*Politecnico di Torino*

# FONDAMENTI DI INFORMATICA

quaderno di testo

Prof. Antonio Lioy

Anno Accademico 2008-2009

*Riproduzione totale o parziale vietata senza il consenso scritto dell'autore.*

# PREFAZIONE

Il presente volumetto raccoglie in forma cartacea il materiale elettronico sviluppato per una serie di corsi di *Fondamenti di Informatica* del Politecnico di Torino.

Più che costituire un “libro di testo”, questo vuole essere una sorta di “quaderno di testo”, in cui gli studenti trovano pre-organizzata una gran parte del materiale presentato in aula: definizioni, schemi, formule, grafici, tabelle. Tradizionalmente questo richiede molto tempo per essere trascritto sui quaderni, senza contare i possibili e probabili errori di trascrizione. Grazie al “quaderno di testo” lo studente può concentrarsi maggiormente sui concetti che vengono esposti e sul filo logico che lega i vari argomenti tra di loro. Nella speranza che questo materiale possa essere utile al maggior numero possibile di studenti, ho scelto di contenere al massimo il costo della pubblicazione ed ho organizzato il materiale in un formato con ampi spazi bianchi per gli appunti personali.

Correzioni e suggerimenti per il miglioramento di questo quaderno possono essermi indirizzati presso il Dipartimento di Automatica e Informatica, oppure inviatimi tramite posta elettronica all’indirizzo `lioy@polito.it`

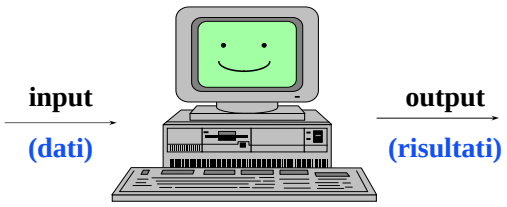
Torino, settembre 2008

L’autore

INDICE

I. L’elaboratore elettronico ..... 1
II. Elaborazione dell’informazione numerica ..... 7
III. Elaborazione dell’informazione non numerica ..... 39
IV. La logica degli elaboratori elettronici ..... 51
V. Tecnologia elettronica digitale ..... 60
VI. Architettura di un elaboratore elettronico ..... 76
VII. Dispositivi di memoria di massa ..... 105
VIII. Dispositivi di input / output ..... 118
IX. Il software ..... 140
X. La programmazione ..... 152
XI. Il sistema operativo ..... 163
XII. Collegamento di dispositivi periferici ..... 175
XIII. Trasmissione dati ..... 188
Appendice. Il codice ASCII ..... 193

## L'elaboratore elettronico (computer)




**processing**
**(elaborazione)**

## Problematiche

- ***codifica dei dati***
  **in un formato comprensibile all'elaboratore**
- ***codifica degli ordini***
  **sequenza di operazioni che compongono l'elaborazione**
- ***decodifica dei risultati***
  **in un formato comprensibile agli umani**

## Hardware

**Definizione:**

***l'insieme delle apparecchiature (elettroniche, meccaniche, ecc.) che costituiscono fisicamente il sistema di elaborazione.***

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Software

**Definizione:**

***l'insieme dei programmi e dei dati che permettono lo svolgimento delle funzioni di elaborazione***

## Istruzioni e programmi

**Un'*istruzione* è la specifica di una operazione che può essere svolta da un elaboratore.**

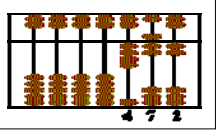
**Un *programma* è un insieme ordinato di istruzioni (non è necessariamente una sequenza!)**

## Cenni storici

**Problema: *effettuare in modo rapido e preciso calcoli matematici complessi*.**

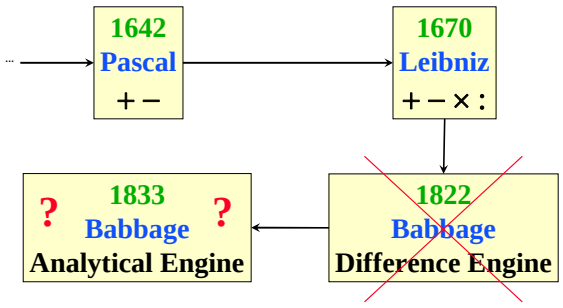
**Soluzioni primitive:**

- **abaco (2000 A.C.)**
- **tavole dei logaritmi (1600)**
- **olio di gomito!**



*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il calcolo meccanico




## L'era elettromeccanica

**1890: censimento USA con lettura elettrica di schede perforate**

**1896: Hollerith fonda la Tabulating Machine Corporation (che nel 1924 diventa IBM = International Business Machines)**

**1944: Aiken costruisce il Mark I**

## L'era elettronica

**1944: Colossus (Bletchley Park, UK)**
**→ tubi a vuoto, decifrazione codici segreti nazisti**
**→ 10 esemplari, distrutti alla fine della guerra**

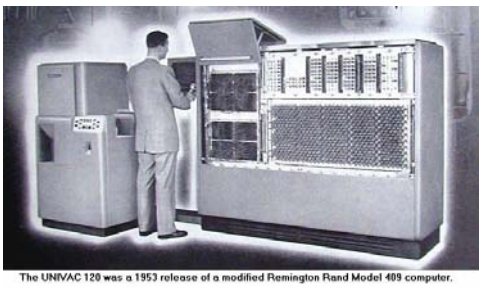
**1946: ENIAC (Eckert e Mauchly, Univ. di Pennsylvania)**
**→ 18000 tubi a vuoto + 1500 relais**

**1951: UNIVAC (Eckert e Mauchly, E & M Computer Corporation)**
**→ primo computer programmabile commerciale**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## ENIAC

## UNIVAC



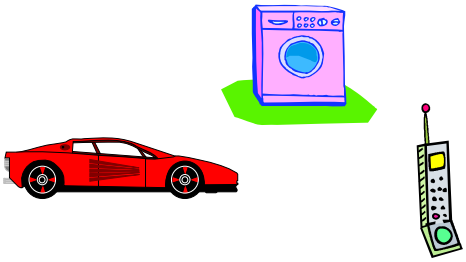
The UNIVAC 120 was a 1953 release of a modified Remington Rand Model 409 computer.

## Le generazioni informatiche

| | | |
|---|---|---|
| **1942-'57,** | **1ª gen.** | **= tubi a vuoto** |
| **1958-'63,** | **2ª gen.** | **= transistori** |
| **1964-'80,** | **3ª gen.** | **= circuiti integrati** |
| **1980-oggi,** | **4ª gen.** | **= circuiti VLSI** |
| **(futuro)** | **5ª gen.** | **= ?** |

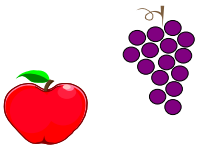
*© A.Lioy - Politecnico di Torino (1992-2008)*

## I più recenti modelli di computer



## I dati

$$V = \frac{4}{3} \cdot \pi \cdot r^3$$



*Nel mezzo del cammin di nostra vita ...*

## I dati numerici

- **sono quelli più usati in ambito scientifico**
- **nei moderni sistemi di elaborazione … tutti gli altri tipi di dato sono trasformati in dati numerici**
- **tutti i tentativi di elaborare direttamente dati non numerici o sono falliti o si sono mostrati molto più inefficienti che non effettuare l'elaborazione solo dopo aver trasformato I dati in forma numerica**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Organismi di standardizzazione

**ANSI ( www.ansi.org )**
**American National Standards Institute**

**CEN ( www.cenorm.be )**
**Comité Européen de Normalisation**

**ETSI ( www.etsi.org )**
**European Telecommunications Standards Institute**

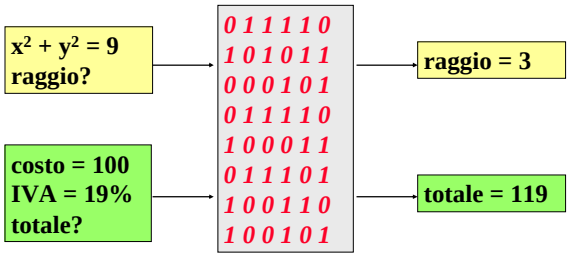
**IEEE ( www.ieee.org )**
**Institute of Electrical and Electronic Engineers**

**IETF ( www.ietf.org )**
**Internet Engineering Task Force**

**ISO ( www.iso.org )**
**International Organization for Standardization**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Elaborazione dell'informazione numerica




## Come contiamo?

$252 = 2 \cdot 100 + 5 \cdot 10 + 2 \cdot 1$

$= 2 \cdot 10^2 + 5 \cdot 10^1 + 2 \cdot 10^0$

**Sistema di numerazione del mondo occidentale (*sistema arabo*):**

- **decimale**
- **posizionale**

## Sistemi di numerazione

**Non posizionali:**

- **numeri romani (es. V, L, D)**
- **operazioni aritmetiche difficili**

**Posizionali:**

- **arabo (decimale)**
- **maya (ventesimale)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Sistema di numerazione posizionale in base B

**Caratteristiche:**

- **cifre: { 0, 1, 2, ..., B-1 }**
- **peso della cifra *i*-esima: $B^i$**
- **rappresentazione (numeri naturali) su N cifre**

$$A = \sum_{i=0}^{N-1} a_i \cdot B^i$$

## Il sistema binario

- **base: 2**
- **cifre: { 0, 1 }**

**Esempio:**

$101_2 = 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$
$= 1 \cdot 4 + 1 \cdot 1$
$= 5_{10}$

## Alcuni numeri binari

| | | | | | |
|---|---|---|---|---|---|
| 0 | ... | 0 | 1000 | ... | 8 |
| 1 | ... | 1 | 1001 | ... | 9 |
| 10 | ... | 2 | 1010 | ... | 10 |
| 11 | ... | 3 | 1011 | ... | 11 |
| 100 | ... | 4 | 1100 | ... | 12 |
| 101 | ... | 5 | 1101 | ... | 13 |
| 110 | ... | 6 | 1110 | ... | 14 |
| 111 | ... | 7 | 1111 | ... | 15 |

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Alcune potenze di due

| | | | | | |
|---|---|---|---|---|---|
| $2^0$ | ... | 1 | $2^9$ | ... | 512 |
| $2^1$ | ... | 2 | $2^{10}$ | ... | 1024 |
| $2^2$ | ... | 4 | $2^{11}$ | ... | 2048 |
| $2^3$ | ... | 8 | $2^{12}$ | ... | 4096 |
| $2^4$ | ... | 16 | $2^{13}$ | ... | 8192 |
| $2^5$ | ... | 32 | $2^{14}$ | ... | 16384 |
| $2^6$ | ... | 64 | $2^{15}$ | ... | 32768 |
| $2^7$ | ... | 128 | $2^{16}$ | ... | 65536 |
| $2^8$ | ... | 256 | | | |

## Terminologia

**BIT (BInary digIT)** 0 1

**BYTE = otto bit** 00110110

**WORD = n byte**

00001111
10101010

## Terminologia

1 0 1 1 0 1 1 0

**MSB** Most Significant Bit

**LSB** Least Significant Bit

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Multipli binari

**Si usano le potenze di due invece di quelle di dieci (potenziale confusione, anche commerciale):**

| | | | |
|---|---|---|---|
| **chilo** | **K** | $2^{10}$ | **~ un migliaio** |
| **mega** | **M** | $2^{20}$ | **~ un milione** |
| **giga** | **G** | $2^{30}$ | **~ un miliardo** |
| **tera** | **T** | $2^{40}$ | **~ mille miliardi** |

**Un tentativo (IEEE) di usare due prefissi diversi (es. K = $10^3$, Ki = $2^{10}$) è largamente fallito.**

## Il sistema ottale

- **base: 8**
  **(talvolta indicata con Q per Octal)**
- **cifre: { 0, 1, 2, 3, 4, 5, 6, 7 }**
- **utile per scrivere in modo compatto i numeri binari ( 3:1 )**

$$\underbrace{1\ 0}_{2}\ \underbrace{1\ 1\ 1}_{7}\ \underbrace{0\ 0\ 1}_{1}\ _2 \qquad 2\ 7\ 1\ _8$$

## Il sistema esadecimale

- **base: 16**
  **(talvolta indicata con H per Hexadecimal)**
- **cifre: { 0, 1, ..., 9, A, B, C, D, E, F }**
- **utile per scrivere in modo compatto i numeri binari ( 4:1 )**

$$\underbrace{1\ 0\ 1\ 1}_{B}\ \underbrace{1\ 0\ 0\ 1}_{9}\ _2 \qquad B\ 9\ _{16}$$

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Limiti del sistema binario

**Considerando numeri naturali da N bit:**

- **combinazioni distinte**
  $2^N$ **(es. 2 bit → 4 simboli)**
- **intervallo di valori**
  $0 \leq x \leq 2^N - 1$
  **(000...0)** $\leq x \leq$ **(111...1)**

## Limiti del sistema binario (rappresentazione naturale)

| *bit* | *simboli* | *min* | *max* |
|---|---|---|---|
| 4 | 16 | 0 | 15 |
| 8 | 256 | 0 | 255 |
| 16 | 65,536 | 0 | 65,535 |
| 32 | 4,294,967,296 | 0 | 4,294,967,295 |

## Conversione di numeri naturali da binario a decimale

**Si applica direttamente la definizione effettuando la somma pesata delle cifre binarie:**

$$1101_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^0$$
$$= 8 + 4 + 1$$
$$= 13_{10}$$

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Corrispondenza decimale/binario

**Regola:**

***per rappresentare K oggetti distinti occorre un numero di bit pari ad almeno***

$$N = \lceil \log_2 K \rceil$$

## La funzione *ceiling*

**La funzione matematica *ceiling* applicata all'argomento *x* restituisce il minor numero intero maggiore o uguale a *x*.**

**In pratica è l'approssimazione per eccesso.**

**Esempi:**

- $\lceil 4 \rceil = 4$
- $\lceil 4.1 \rceil = 5$
- $\lceil -4 \rceil = -4$
- $\lceil -4.1 \rceil = -4$

## Calcolo del logaritmo in base 2

- **avendo a disposizione una calcolatrice che calcola il logaritmo naturale (ln) oppure quello in base 10 (Log)**

$$\log_2 x = \text{Log } x \,/\, \text{Log } 2 = \ln x \,/\, \ln 2$$

© *A.Lioy - Politecnico di Torino (1992-2008)*

## Corrispondenza decimale/binario

**Regola pratico-empirica (ma imprecisa …):**

***in media, occorrono 3.25 bit per rappresentare una cifra decimale***

**Esempi:**

$xx_{10} \rightarrow bbbbbbb_2$

$xxx_{10} \rightarrow bbbbbbbbbb_2$

## Conversione di numeri naturali da decimale a binario

$A_2 = a_{N-1} \cdot 2^{N-1} + ... + a_2 \cdot 2^2 + a_1 \cdot 2^1 + a_0$

$= 2\,(a_{N-1} \cdot 2^{N-2} + ... + a_2 \cdot 2^1 + a_1) + a_0$

$= 2\,[\,2\,(a_{N-1} \cdot 2^{N-3} + ... + a_2) + a_1] + a_0$

$A_{10} = 2 \cdot Q_0 + R_0$

$= 2 \cdot (\,2 \cdot Q_1 + R_1\,) + R_0$

## Da decimale a binario

**Dal confronto tra l'espressione di A come numero binario e come risultato della divisione intera per due si ottiene:**

- $a_0 = R_0$
- $a_1 = R_1$
- ...
- $a_i = R_i$

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Da decimale a binario

**Regola pratica:**

- **divisioni successive per due**
- **si prendono i resti in ordine inverso**

| | | | | | |
|---|---|---|---|---|---|
| **13** | **6** | **3** | **1** | **0** | ***quozienti*** |
| | **1** | **0** | **1** | **1** | ***resti*** |

**$13_{10} = 1101_2$**

## Esercizi

**Convertire da decimale a binario, e viceversa:**

- **$14_{10}$** **[ $1110_2$ ]**
- **$33_{10}$** **[ $100001_2$ ]**
- **$57_{10}$** **[ $111001_2$ ]**
- **$31_{10}$** **[ $11111_2$ ]**

**Convertire in binario e decimale:**

- **$123_8$** **[ $1010011_2$ ; $83_{10}$ ]**
- **$128_8$** **[ impossibile ]**
- **$A1A_{16}$** **[ $1010\ 0001\ 1010_2$ ; $2586_{10}$ ]**
- **$123_H$** **[ $0001\ 0010\ 0011_2$ ; $291_{10}$ ]**

## Esercizi

- **determinare la base X in cui vale la seguente uguaglianza:**
  **$43_X + 24_X = 100_X$** **[ x = 7 ]**

- **effettuare i seguenti cambiamenti di base:**
  - **$23_{10}$ in base 6** **[ $35_6$ ]**
  - **$11_7$ in base 10** **[ $8_{10}$ ]**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Operazioni aritmetiche

## Somma in binario

**Regole base:**

**0 + 0 = 0**
**0 + 1 = 1**
**1 + 0 = 1**
**1 + 1 = 0** **( carry = 1 )**

## Somma in binario

**Si effettuano le somme parziali tra i bit dello stesso peso, propagando gli eventuali riporti:**

```
  1 1
  0 1 1 0 +
  0 1 1 1 =
-----------
  1 1 0 1
```

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Overflow

**Si usa il termine *overflow* per indicare l'errore che si verifica in un sistema di calcolo automatico quando il risultato di un'operazione non è rappresentabile con la medesima codifica e numero di bit degli operandi.**

## Overflow

**Nella somma in binario puro si ha overflow quando:**

- **si lavora con numero fisso di bit**
- **si ha carry sul MSB**

## Overflow - esempio

***Ipotesi*: operazioni su numeri da 4 bit codificati in binario puro**

```
          0101 +
          1110 =
        -------
overflow → 10011
```

© *A.Lioy - Politecnico di Torino (1992-2008)*

## Sottrazione in binario

**Regole base:**

**0 - 0 = 0**
**0 - 1 = 1** **( borrow = 1 )**
**1 - 0 = 1**
**1 - 1 = 0**

## Sottrazione in binario

**Si effettuano le differenze parziali tra i bit dello stesso peso, propagando gli eventuali prestiti:**

```
  0 2
  0 1 2
  1 0 0 1 –
  0 1 1 1 =
  ---------
  0 0 1 0
```

## Overflow

**Nella sottrazione in binario puro si ha overflow quando:**

- **si lavora con numero fisso di bit**
- **si ha borrow sul MSB**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Overflow - esempio

***Ipotesi*: operazioni su numeri da 4 bit codificati in binario puro**

**overflow →**

```
 10101 –
  1100 =
-------
  1001
```

## Moltiplicazione e divisione in binario

**In linea teorica è possibile operare come nel sistema decimale.**

**Nella pratica spesso si usano accorgimenti particolari basati sull'operazione di *shift*.**

## L'operazione di shift

- **consiste nel "far scorrere" i bit (a sinistra o a destra) inserendo opportuni valori nei posti lasciati liberi**
- **equivale ad una moltiplicazione o divisione per una potenza di due**
- **errori: overflow e troncamento**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Shift a sinistra

- **si inserisce come LSB un bit a zero**
- **equivale ad una moltiplicazione per due**

- **esempi (binario puro su 4 bit):**

  **0011 « 1 = 0110** **( 3 × 2 = 6 )**

  **0011 « 2 = 1100** **( 3 × 4 = 12 )**

  **0011 « 3 = 1000** **overflow!**

## Moltiplicazione / shift

**E' possibile eseguire un qualunque prodotto usando solo operazioni di shift e somma:**

$34 \times 7 = 34 \times ( 4 + 2 + 1 )$

$= 34 \times 4 + 34 \times 2 + 34 \times 1$

$= 34 \text{ « } 2 + 34 \text{ « } 1 + 34$

## Shift a destra

- **equivale ad una divisione per due**
- **shift logico: inserisce come MSB un bit a zero**
- **shift aritmetico: inserisce un bit uguale al MSB**

- **esempi (binario puro su 4 bit):**

  **0110 » 1 = 0011** **( 6 : 2 = 3 )**

  **0110 » 2 = 0001** **( 6 : 4 = 1 )**
  **troncamento!**

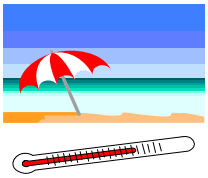
*© A.Lioy - Politecnico di Torino (1992-2008)*

## Esercizi

**Trasformare gli operandi in binario (codifica naturale su 5 bit) e quindi eseguire le seguenti operazioni, segnalando eventuali errori:**

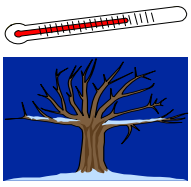
- **23 + 12 6 + 7**
- **23 – 12 6 – 7**
- **5 << 2 5 << 4**
- **17 >> 2 17 >> 3 (shift logico)**

## I numeri relativi



**+ 32 °C**

**-5 °C**



## I numeri con segno

**Il segno dei numeri può essere solo di due tipi:**

**positivo ( + )**

**negativo ( – )**

**E' quindi facile rappresentarlo in binario ... ma non sempre la soluzione più semplice è quella migliore!**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Codifica "modulo e segno"

- **un bit per il segno:**
  - **0 = segno positivo ( + )**
  - **1= segno negativo ( – )**
- **N-1 bit per il valore assoluto (anche detto *modulo*)**

| *segno* | *modulo* |
|---|---|
| **1 bit** | **N-1 bit** |

## Modulo e segno: esempi

**Usando una codifica su quattro bit:**

$+ 3_{10} \rightarrow 0011_{M\&S}$

$- 3_{10} \rightarrow 1011_{M\&S}$

$0000_{M\&S} \rightarrow + 0_{10}$

$1000_{M\&S} \rightarrow - 0_{10}$

## Modulo e segno

**Svantaggi:**

- **doppio zero (+ 0, - 0)**
- **operazioni complesse**
- **es. somma A+B**

| | A > 0 | A < 0 |
|---|---|---|
| B > 0 | A + B | B - \|A\| |
| B < 0 | A - \|B\| | - ( \|A\| + \|B\| ) |

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Modulo e segno: limiti

**In una rappresentazione su N bit:**

$$-(2^{N-1}-1) \leq x \leq +(2^{N-1}-1)$$

**Esempi:**

- **8 bit = [ -127 ... +127 ]**
- **16 bit = [ -32,767 ... +32,767 ]**

## Complemento a uno (operazione)

**Considerando numeri binari da N bit, si definisce *complemento a uno* di un numero A la quantità:**

$$\overline{A} = 2^N - 1 - A$$

**Viene anche detto semplicemente *complemento*.**

**Si indica con una linea orizzontale sopra al numero oppure con un apostrofo (A').**

## Complemento a uno

**Regola pratica:**

***il complemento a uno di un numero binario A si ottiene cambiando il valore di tutti i suoi bit***

**Esempio:**

**A = 1011 → $\overline{A}$ = 0100**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Complemento a due (operazione)

**Considerando numeri binari da N bit, si definisce *complemento a due* di un numero A la quantità:**

$$\overline{\overline{A}} = 2^N - A = \overline{A} + 1$$

## Complemento a due

**Regola pratica:**

***il complemento a due di un numero binario A si ottiene sommando uno al suo complemento***

**Esempio:**

**A = 1011, $\overline{A}$=0100 → $\overline{\overline{A}}$ = 0101**

## Complemento a due (bis)

**Regola pratica (bis):**

***il complemento a due di un numero binario A si ottiene partendo dal LSB e copiando tutti i bit sino al primo "1" incluso e complementando tutti i bit successivi.***

**Esempio:**

**A = 10110 → $\overline{\overline{A}}$ = 01010**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Codifica in complemento a due

**Per rappresentare numeri relativi:**

**( A ≥ 0 )** $A_{CA2} = 0\ A_2$

**( A < 0 )** $A_{CA2} = \overline{\overline{|A|_2}}$

**Così facendo, l'MSB indica sempre il segno ( 0 = +, 1 = – )**

## Complemento a due: esempi

**Usando una codifica su quattro bit:**

$+ 3_{10} \rightarrow 0\ (3_2) \rightarrow 0011_{CA2}$

$- 3_{10} \rightarrow \overline{0011} + 1 \rightarrow 1101_{CA2}$

$0000_{CA2} \rightarrow + (000) \rightarrow + 0_{10}$

$1000_{CA2} \rightarrow -\overline{(1000 - 1)} \rightarrow - 8_{10}$

## CA2: esempio con passaggi

**Quesito:**

- **1000 (CA2 su 4 bit) = x (base 10)?**

**Passaggi:**

- **1000 – 1 = 0111**
- **(0111)' = 1000**
- **1000 (base 2) = 8 (base 10)**

**Risposta:**

- **x = – 8**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Esercizi

**Trasformare i seguenti numeri da (oppure a) codifica modulo e segno e complemento a due su 5 bit:**

- **+11, – 11**
- **+13, – 13**
- **01010, 11010**

## Complemento a due: limiti

**In una rappresentazione su N bit:**

$$-2^{N-1} \leq x \leq +(2^{N-1}-1)$$

**Esempi:**

- **8 bit = [ -128 ... +127 ]**
- **16 bit = [ -32,768 ... +32,767 ]**

## Somma e sottrazione in CA2

**La somma si effettua direttamente, senza badare ai segni degli operandi:**

$$A_{CA2} + B_{CA2} \rightarrow A_{CA2} + B_{CA2}$$

**La sottrazione si effettua sommando al minuendo il CA2 del sottraendo:**

$$A_{CA2} - B_{CA2} \rightarrow A_{CA2} + \overline{\overline{B_{CA2}}}$$

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Sottrazione in CA2 - esempio

**00100110 – 11001011**

**00100110 +**
**00110101 =**
**----------**
**01011011**

**verifica: 38 – (– 53) = 91**

## Shift in complemento a due

- **quando si effettua lo shift a destra di un numero in complemento a due, si fa sempre lo shift aritmetico**
- **esempio (CA2 su 4 bit):**
  **– 5 » 1**
  **1011 » 1 = 1101 (con troncamento)**
  **verifica: $1101_{CA2} = 0011 = -3_{10}$**

## Shift in complemento a due

- **quando si effettua lo shift a sinistra di un numero in complemento a due, si inserisce a destra uno zero ma si deve prestare attenzione all'overflow**
- **esempio (CA2 su 4 bit):**
  **+5 « 1**
  **ossia**
  **0101 « 1 = 1010 (overflow!)**
  **verifica:**
  **$0101_{CA2} = +5_{10}$**
  **$1010_{CA2} = -6_{10}$**

*© A.Lioy - Politecnico di Torino (1992-2008)*

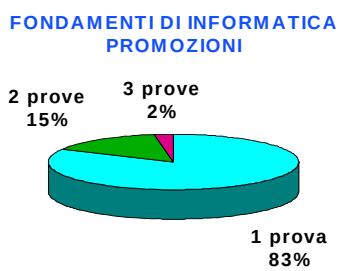
## Overflow nella somma in CA2

- ***Operandi con segno discorde*: non si può mai verificare overflow**
- ***Operandi con segno concorde*: c'è overflow quando il risultato ha segno discorde**
- **In ogni caso, si trascura sempre il carry sul MSB**

## Overflow in CA2

| | |
|---|---|
| **0101 +** | **1110 +** |
| **0100 =** | **1101 =** |
| --- | --- |
| **1001** | **11011 =** |
| | **1011** |
| ***overflow!*** | ***OK*** |

## I numeri razionali (o frazionari)




*© A.Lioy - Politecnico di Torino (1992-2008)*

## Numeri razionali

**Esprimibili come un rapporto di numeri interi.**

**In altre parole sono il risultato di una divisione tra numeri interi.**

**Rappresentati in forma frazionaria oppure con:**

- **parte intera**
- **parte frazionaria**

**Ad esempio:**

$$R = 15 / 4 = 3.75$$

## Conversione in binario della parte frazionaria di un numero

$$A = 0, \ldots$$
$$= a_{-1}\, 2^{-1} + a_{-2}\, 2^{-2} + \ldots$$
$$2 \times A = a_{-1}\, 2^{0} + a_{-2}\, 2^{-1} + a_{-3}\, 2^{-2} + \ldots$$
$$= a_{-1} + ( a_{-2}\, 2^{-1} + a_{-3}\, 2^{-2} + \ldots )$$

**Si ottiene quindi:**

$$a_{-1} = int ( 2 \times A )$$
$$a_{-2} = int ( 2 \times frac(2 \times A) )$$
$$a_{-3} = int ( \ldots )$$

## Parte frazionaria

**Regola pratica:**

- **moltiplicazioni successive per due**
- **si considerano le parti intere nell'ordine in cui sono state generate**

| .25 | .5 | .0 |
|---|---|---|
| | 0 | 1 |

$$( 0.25 )_{10} = ( 0.01 )_{2}$$

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Numeri non rappresentabili

**Esistono numeri decimali frazionari finiti non rappresentabili *esattamente* in binario (numero infinito di cifre):**

| .4 | .8 | .6 | .2 | .4 | ... |
|---|---|---|---|---|---|
| | 0 | 1 | 1 | 0 | ... |

$$(0.4)_{10} = (0.\overline{0110})_2$$

***Problema: quante cifre si considerano?***

## Errore assoluto ( ε )

***Dato un numero A composto da N cifre, l'errore assoluto della sua rappresentazione è la quantità - non nulla - più piccola (in valore assoluto) rappresentabile con N cifre.***

## Errore assoluto - esempi

**Qualunque sia la base ed il numero di cifre N, la precisione assoluta dei numeri interi è sempre 1:**

- **es.** $5_{10}$ $\varepsilon = 1_{10}$
- **es.** $27_{10}$ $\varepsilon = 1_{10}$

**La precisione assoluta dei numeri razionali dipende dal numero di cifre usate per rappresentarli:**

- **es.** $0.5_{10}$ $\varepsilon = 0.1_{10}$
- **es.** $0.50_{10}$ $\varepsilon = 0.01_{10}$
- **es.** $1.0_{10}$ $\varepsilon = 0.1_{10}$

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Errore relativo ( η )

***Dato un numero A composto da N cifre, l'errore relativo della sua rappresentazione è il rapporto tra l'errore assoluto ed il valore assoluto del numero A***

$$\eta = \frac{\varepsilon}{|A|}$$

## Errore relativo - esempi

**L'errore relativo della rappresentazione di qualunque numero (sia intero sia razionale) è variabile e dipende sia dal valore che dal numero di cifre usato per la rappresentazione.**

- **es. 100** $\varepsilon = 1$ $\eta = 1\ \%$
- **es. 200** $\varepsilon = 1$ $\eta = 0.5\ \%$

## Approssimazione dei numeri frazionari

**L'errore assoluto è inferiore al peso del LSB:**

$0.01_2 : \varepsilon \leq 1/4$

**Verifica:**

- **limite inferiore (best case)** $0.010000... = 0.25$
- **limite superiore (worst case)** $0.011111... =$
  $= 1/4 + 1/8 + 1/16 + ...$
  $= 1/4 \times ( 1 + 1/2 + 1/4 + ... )$
  $< 0.25 \times 2$
- **intervallo [ 0.25 ... 0.5 [** $< 1/4$

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Approssimazione dei numeri frazionari

**Dato l'errore decimale desiderato, si deve includere nella rappresentazione almeno la cifra binaria di peso immediatamente inferiore**

$0.4_{10}$ (con $\varepsilon \leq 0.1_{10}$) = $0.0110_2$

( $1/16 < 0.1_{10}$ )

**Verifica:**

- $0.0110_2 = 0.375_{10}$
- $|\ 0.4 - 0.375\ | = 0.025 < 0.1$

## Esercizi

**Trasformare i seguenti numeri decimali in binario puro con errore assoluto inferiore a 1/100:**

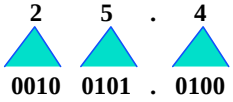
- $0.30_{10}$ $0.70_{10}$ $0.50_{10}$

**Trasformare i seguenti numeri binari in decimale, indicando la loro precisione assoluta:**

- $0.0011_2$ $0.1010000_2$ $0.101_2$

## Codice BCD (Binary-Coded Decimal)

**Si codificano in binario su quattro bit le singole cifre decimali:**




( 25.4 )$_{10}$ = ( 0010 0101 . 0100 )$_{BCD}$

© *A.Lioy - Politecnico di Torino (1992-2008)*

## Codice BCD

+ **nessun errore di conversione**
+ **stessa precisione dei calcoli decimali**
– **spreco di bit rispetto alla notazione binaria pura (1010 ... 1111 non usati)**
– **operazioni lente (sono decimali!)**

***Talvolta usato nei programmi contabili***

## Numeri reali

**Problematiche:**

- **intervallo di valori rappresentabile**
- **precisione della rappresentazione**
  - **errore assoluto**
  - **errore relativo**

## Rappresentazione fixed-point

**Si usa un numero fisso di bit per la parte intera e per quella frazionaria (e non si rappresenta la virgola!)**

**Ad esempio (4 + 4 bit, binario puro):**

**15.9375 = 1111 1111**

**0.0625 = 0000 0001**

**virgola sottintesa**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Fixed-point

+ **operandi allineati = operazioni facili**
+ **precisione assoluta fissa**
– **piccolo intervallo di valori**
– **divisione tra parte intera e parte frazionaria fissa e non adatta a tutti i campi applicativi**
– **precisione relativa variabile**

**Il fixed-point viene quindi usato raramente nei sistemi general-purpose, mentre è più frequente nei sistemi special-purpose.**

## Codifica fixed-point in CA2

- **si consideri il numero A = I.F ove I = *int* ( A ) e F = *fraz* ( A )**
- **se $A \geq 0$ → $A_{CA2} = 0\ I_2 . F_2$**
- **se $A < 0$ → $A_{CA2} = \overline{I_2 . F_2} + 1_{LSB}$**

## Limiti della codifica fixed-point CA2

**Con N bit per la parte intera e F bit per la parte frazionaria:**

- ***frazmax* = $1/2 + 1/4 + ... + 1/2^F$**
- **max = $+ ( 2^{N-1} - 1 )$ . *frazmax***
- **min = $- 2^{N-1}$ . 0**
- **risoluzione (o *step*) = $1/2^F$**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Limiti della codifica fixed-point CA2

**Esempio (fixed-point, CA2, 3 I + 2 F):**

- **max = 011 11 = + 3 . 75**
- **min = 100 00 = – 4 . 0**
- **step = 000 01 = 0.25**
- **valori**
  **–4.0 –3.75 –3.5 –3.25**
  **–3.0 –2.75 –2.5 . . .**
  **. . . . . . . . . +2.75**
  **+3.0 +3.25 +3.5 +3.75**

## Rappresentazione floating-point

**Si usa sempre il formato esponenziale:**

$$N = \text{mantissa} \cdot \text{base}^{\text{esponente}}$$

**Vari formati decimali:**

- **esempi di possibili codifiche per N=35000**
- **standard $3.5 \cdot 10^4$ 3.5E+4**
- **scientifico $0.35 \cdot 10^5$ 0.35E+5**
- **ingegneristico $35 \cdot 10^3$ 35E+3**

## Floating-point

+ **grande intervallo di valori**
+ **errore relativo fisso**
– **operandi non allineati**
  **→ operazioni complesse**
– **errore assoluto variabile**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Formato IEEE-754

**Standard IEEE per il floating-point:**

- **mantissa nella forma "1, ..."**
- **"1," sottinteso**
- **mantissa in modulo e segno**
- **base 2**
- **esponente in codice *eccesso K***

## Codice eccesso K

**Si rappresentano i numeri interi relativi sommando a ciascuno una quantità fissa K (*offset* o *bias*).**

**Esempio (codice eccesso 4 su 3 bit):**

| | |
|---|---|
| **– 4 ... 0** | **0 ... 4** |
| **– 3 ... 1** | **+ 1 ... 5** |
| **– 2 ... 2** | **+ 2 ... 6** |
| **– 1 ... 3** | **+ 3 ... 7** |

## Singola precisione (SP)

**IEEE-754 usa 32 bit con l'esponente in codice eccesso 127:**

| *segno* | *esponente* | *mantissa* |
|---|---|---|
| 1 bit | 8 bit | 23 bit |

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Doppia precisione (DP)

**IEEE-754 usa 64 bit con l'esponente in codice eccesso 1023:**

| *segno* | *esponente* | *mantissa* |
|---|---|---|
| 1 bit | 11 bit | 52 bit |

## IEEE-754 – codifiche speciali

| E | M | valore |
|---|---|---|
| 0 | 0 | ±0 |
| 0 | ≠ 0 | non normalizzato |
| *max* | 0 | ±∞ |
| *max* | ≠ 0 | NaN (Not a Number) |

## IEEE-754 SP

$M_{min} = 1$ $M_{max} < 2$

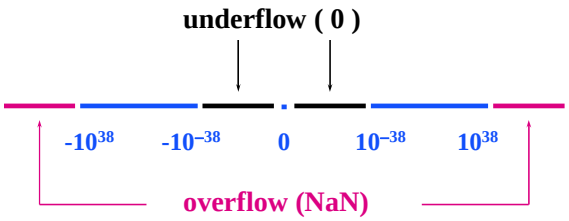
$E_{min} = -126$ $E_{max} = +127$

**Valori rappresentabili:**

- **zero**
- **$( 2^{-126} \ldots 2^{127} ) \sim ( 10^{-38} \ldots 10^{38} )$**
- **$(- 2^{127} \ldots - 2^{-126} ) \sim (- 10^{38} \ldots - 10^{-38})$**

© *A.Lioy - Politecnico di Torino (1992-2008)*

## IEEE-754 SP: intervallo di valori




## IEEE-754: cifre significative

**Singola precisione:**

- **|M| = 24 bit → 7 cifre decimali**

**Doppia precisione:**

- **|M| = 53 bit → 16 cifre decimali**

## Conversione da decimale a IEEE-754 SP

***Convertire –23.625 in IEEE-754 SP:***

**S = 1**

**$23.625_{10} = 10111.101_2 = 1.0111101_2 \cdot 2^4$**

**M = 01111010...0**

**E = 4 + 127 = 131 = 10000011**

**1 10000011 01111010000000000000000**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Conversione da IEEE-754 SP a decimale

*0 01111110 00100000000000000000000*

**S = 0 → segno positivo**

**E = $01111110_2$ = 126 - 127 = - 1 → $2^{-1}$**

**M = 1.$00100000000000000000000_2$**

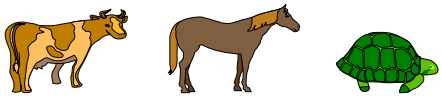
**A = + $1.001_2 \cdot 2^{-1}$ = + $0.1001_2$ = + 0.5625000**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Elaborazione dell'informazione non numerica

## Informazione non numerica

**Se in quantità finita, si può mettere in corrispondenza coi numeri interi.**



**00** **01** **10**

## Dagli oggetti ai codici

**Per rappresentare K oggetti distinti occorre un numero minimo di bit pari a:**

$$N = \lceil \log_2 K \rceil$$

**$\lceil x \rceil$ è la funzione *ceiling*: il più piccolo numero intero maggiore o uguale a quello dato**

**$\lceil 3.9 \rceil = 4$ $\lceil 4.0 \rceil = 4$ $\lceil -3.9 \rceil = -3$**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Caratteri

**Occorre una codifica standard perché è il genere di informazione più scambiata:**

- **codice ASCII (American Standard Code for Information Interchange)**
- **codice EBCDIC (Extended BCD Interchange Code)**

## Codice ASCII

**Usato anche nelle telecomunicazioni.**

**Usa 7 bit (8 bit) per rappresentare:**

- **52 caratteri alfabetici (`a...z A...Z`)**
- **10 cifre (`0...9`)**
- **segni di interpunzione (`, ; ! ? ...`)**
- **caratteri di controllo**

## Caratteri di controllo

| | | | |
|---|---|---|---|
| | CR | ( 13 ) | Carriage Return |
| LF, | NL | ( 10 ) | New Line, Line Feed |
| FF, | NP | ( 12 ) | New Page, Form Feed |
| | HT | ( 9 ) | Horizontal Tab |
| | VT | ( 11 ) | Vertical Tab |
| | NUL | ( 0 ) | Null |
| | BEL | ( 7 ) | Bell |
| | EOT | ( 4 ) | End-Of-Transmission |
| | ... | ... | ... |

© A.Lioy - Politecnico di Torino (1992-2008)

## Codice ASCII - esempio

| | | | |
|---|---|---|---|
| 01000001 | A | 00100000 | |
| 01110101 | u | 01110100 | t |
| 01100111 | g | 01110101 | u |
| 01110101 | u | 01110100 | t |
| 01110010 | r | 01110100 | t |
| 01101001 | i | 01101001 | i |
| 00100000 | | 00100001 | ! |
| 01100001 | a | | |

## Rappresentazione di un testo in formato ASCII

- caratteri in codice ASCII
- ogni riga terminata dal ***terminatore di riga***:
  - in MS-DOS e Windows = CR + LF
  - in UNIX = LF
  - in MacOS = CR
- pagine talvolta separate da FF

## Codice ASCII: problema

**Un testo è composto di 40 pagine, ciascuna contenente 50 righe di 60 caratteri.**

**Se il testo viene codificato in ASCII, con righe terminate da CR-LF e pagine terminate da FF, quanti byte sono necessari per memorizzare il testo?**

**riga: 60 caratteri + CR + LF = 62 byte**

**pagina: 50 × 62 + FF = 3101 byte**

**testo: 40 × 3101 = 124,040 byte**
**testo: 124,040 : 1024 = 121.1 KB**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Caratteri estesi

**I caratteri con codice ASCII maggiore di 127 non sono standard (perché non presenti nel codice ASCII originale che era su soli 7 bit).**

**In pratica i caratteri con MSB=1 sono usati per:**

- **codici di controllo**
- **alfabeti locali (es. à á ä ã æ )**
- **caratteri grafici (es. ♣ ♦♥♠ ∝ ⇔ √ )**

## Codifiche ASCII estese

**Telecomunicazioni:**

- **MSB è bit di parità**

**Personal Computer:**

- ***code page (CP)* nazionali**
- **es. MS-DOS CP 850 (138 = è, 232 = Þ)**
- **es. MS-Windows CP 1252 (138 = Š, 232 = è)**

**Alfabeti con più di 256 simboli:**

- **DBCS (Double Byte Character Set)**
- **Unicode (codice a 16 bit) UTF-8, UTF-7**

## Alfabeti ISO

**ISO-8859-1 (Latin-1) = europa occidentale**
**ISO-8859-2 (Latin-2) = europa centrale**
**ISO-8859-3 (Latin-3)**
**ISO-8859-4 (Latin-4) = paesi baltici**
**ISO-8859-5 (Latin/Cyrillic)**
**ISO-8859-6 (Latin/Arabic)**
**ISO-8859-7 (Latin/Greek)**
**ISO-8859-8 (Latin/Hebrew)**
**ISO-8859-9 (Latin-5) = Turchia**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Compressione dei dati

- **per ridurre la quantità di dati da memorizzare / trasmettere si può cercare di eliminare le ripetizioni**
- **esempio (con fattore moltiplicativo) AAAAAAABAAAAAAA può diventare 7xA B 7xA**
- **esempio (con "dizionario") "I Torinesi abitano a Torino" può diventare "I äesi abitano a äo" con l'informazione aggiuntiva (dizionario) ä = Torin**

## Compressione senza / con perdita

- **compressione senza perdita (*lossless*) se è possibile ricostruire esattamente i dati originali**
- **compressione con perdita (*lossy*) se si ricostruiscono dei dati molto simili ma non identici a quelli originali (si trascura qualche dettaglio)**
- **esempio (compressione lossy) AAAAAAABAAAAAAA può diventare 15 x A (trascurando B)**

## Codifica di immagini

- **se l'immagine è molto complessa ed irregolare si preferisce una codifica *raster* (anche detta *bitmap* o *pixmap*), ossia fatta punto per punto**
- **punto = *dot* o *pixel* (picture element)**
- **se l'immagine contiene forme geometriche regolari si preferisce una codifica *vettoriale*, ossia fatta specificando gli elementi geometrici**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Codifica raster

- si suddivide l'immagine in tanti punti, definendo R righe e C colonne
- per ogni punto si memorizza il colore:
  - scelto tra i $2^N$ disponibili nella tavolozza ( *palette* )
  - colore = dato codificato su N bit
- memoria totale per un'immagine:

$$M = R \times C \times N \text{ bit / pixel}$$

## Formati raster

BMP (Bitmap)
- Microsoft
- pixmap, senza compressione

GIF (Graphic Interchange Format):
- 8 bit/pixel (256 colori) → immagini semplici
- buona compressione (senza perdita)
- animazione tramite sequenza

PNG (Portable Network Graphic)
- standard W3C

## Formati raster

JPEG (Joint Picture Experts Group):
- 24 bit/pixel (16M colori) → vasta gamma di colori e sfumature
- grado di compressione variabile
- massima compressione con perdita

© A.Lioy - Politecnico di Torino (1992-2008)

## Codifica vettoriale

- si identificano gli elementi geometrici presenti (linee, cerchi, ...)
- si codificano con un opportuno formato (es. `LINE(0,0,10,10)` )
- formati più comuni:
  - CGM, DWG, DXF
  - PostScript e PDF sono formati misti raster/vettoriale/testo

## Modifica delle dimensioni di un'immagine

- si parla di "scaling": riduzione o ingrandimento
- in generale scalano meglio le immagini vettoriali (=istruzioni per disegnarle) che quelle raster (=riduzione o ingrandimento dei pixel)
- differenza evidente soprattutto negli ingrandimenti

## Scaling: confronto raster - vettoriale

raster

vettoriale

© A.Lioy - Politecnico di Torino (1992-2008)

## Esercizio

**Un calcolatore ha una memoria grafica di 4 MB usata in modo raster senza compressione; quanti colori può visualizzare simultaneamente con una risoluzione di 1024 x 768?**

**Risposta: circa 4000 miliardi di colori**

## Codifica di filmati video

- **filmato = sequenza di immagini (ciascuna chiamata fotogramma o *frame*)**
- **minimo 16 frame/s (FPS) affinché l'occhio umano non percepisca i singoli fotogrammi**
- **è inefficiente codificare completamente ogni frame**
- **alcuni frame si codificano tutti, altri solo nelle parti che differiscono da quelli "adiacenti"**

## Formati video

**MPEG (Moving Picture Experts Group):**
- **standard anche non informatico (SAT, DTT)**
- **molto efficiente ma complesso**

**QuickTime:**
- **proposto da Apple**
- **un po' meno efficiente ma più semplice**

**Indeo / AVI:**
- **proposto da Intel, usato da MS**
- **inefficiente**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## MPEG

**MPEG-1 (1991)**
- **per video-CD e CD-I; 1.5 Mbps**

**MPEG-2 (1994)**
- **per TV digitale (e DVD); 4 … 10 Mbps**

**MPEG-4 (1998)**
- **interattività**
- **compressione (64 Kbps o 2 Mbps)**

**MPEG-7 (in corso di sviluppo)**
- **meta-informazioni su dati multimediali, per indicizzazione e ricerca**

## Formati video TV (analogici)

**PAL (Phase Alteration by Line)**
- **Europa ed Asia**
- **576 righe di 768 pixel, 25 FPS**

**NTSC (National Television Standards Committee)**
- **USA**
- **525 righe di 768 pixel, 30 FPS**

**nota: pellicola cinematografica = 24 FPS**

## Codifica di suoni

- **frequenze udibili < 4 KHz**
- **campionamento ad intervalli regolari (valore tipico 44 KHz)**
- **10 MB = 1' di audio con qualità CD musicale (stereo, 16 bit/campione)**
- **compressione tipica 2:1 (lossless)**
  - **formato WAV (PC MS-Windows)**
  - **formato AU (Next, SUN)**
  - **formato AIFF (Apple, SGI)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il formato MP3

- **MPEG-1 audio layer 3**
- **compressione con perdita, ma la perdita riguarda parti a cui l'orecchio umano è poco sensibile**
- **qualità quasi uguale a CD-audio, con compressione 12x**
- **ottimizzato per velocità 32-320 Kbps (tipicamente, 64 Kbps)**

## Il formato MIDI

**MIDI:**

- **codifica le note e gli strumenti che devono eseguirle**
- **solo musica, non voce umana**
- **per riprodurre i suoni richiede un sintetizzatore o "campioni" dei vari strumenti**
- **molto efficiente (6'19" della 5ª sinfonia di Beethoven = 91 KB)**

## Protezione dagli errori

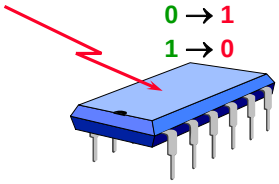
**Esistono fenomeni fisici che possono accidentalmente cambiare l'informazione memorizzata:**

- **esposizione ad alte temperature**
- **radiazioni (es. raggi X, gamma, cosmici)**
- **interferenze elettromagnetiche**
- **campi magnetici**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Tipi di errore

**Nel sistema binario, un errore può essere solo l'inversione di un bit:**




## Codici a rivelazione d'errore

- **aggiungono ai dati dei bit (di controllo) per rilevare errori**
- **in generale, aggiungendo N bit si possono rivelare tutti gli errori che colpiscono al massimo N bit**

## Codici a rivelazione d'errore

**Codice più semplice: parità dispari**

| | |
|---|---|
| 0 0 1 0 1 0 0 | *dato* |
| 1 0 0 1 0 1 0 0 | *dato + bit di parità* |
| *dati osservati* | |
| 1 0 0 1 0 1 0 0 | *nessun errore* |
| 1 1 0 1 0 1 0 0 | *errore* (0 → 1) |
| 1 0 0 1 0 0 0 0 | *errore* (1 → 0) |

© A.Lioy - Politecnico di Torino (1992-2008)

## Codici a correzione d'errore

- **aggiungono ai dati dei bit (di controllo) per rivelare e correggere errori**
- **in generale, con lo stesso codice si possono rivelare un massimo di N errori e correggerne fino a N-1**

## Codici a correzione d'errore

**Esempio: parità orizzontale e verticale (codice di Hamming 2,1)**

| *dati + parità* | *dato osservato* | |
|---|---|---|
| 1 0 0 0 0 | 1 0 0 0 0 | |
| 0 0 0 1 0 | 0 0 1 1 0 | ← *errore!* |
| 1 0 1 0 1 | 1 0 1 0 1 | |
| 1 0 0 1 1 | 1 0 0 1 1 | |
| 0 1 0 1 1 | 0 1 0 1 1 | *bit errato* |
| | | *errore!* |

## Esercizio

**I seguenti numeri sono protetti con un codice di parità dispari ed il bit di parità è il MSB. Indicare quali numeri sono affetti da errore:**

- **00110010**
- **11001010**
- **01000001**
- **11100001**
- **10101000**

**Cosa cambierebbe se il bit di parità fosse il LSB?**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## La logica degli elaboratori elettronici

## La logica Booleana

**Nel 1847 George Boole introdusse un nuovo tipo di logica formale, basata esclusivamente su enunciati di cui fosse possibile verificare in modo *inequivocabile* la verità o la falsità.**

## Variabili Booleane

**Variabili in grado di assumere solo due valori:**

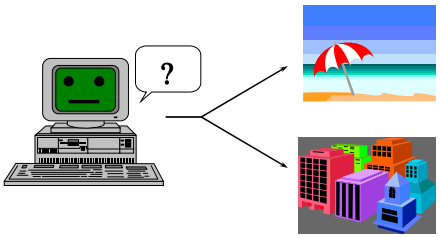
**VERO** **FALSO**

**In ogni problema è importante distinguere le variabili *indipendenti* da quelle *dipendenti*.**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Logica Booleana: un esempio

***Se* domani ci sarà il sole, *allora* andrò a fare una gita.**




## "Se domani ci sarà il sole, allora andrò a fare una gita"

**Variabile indipendente:**
**A** (**"*domani ci sarà il sole*"**)

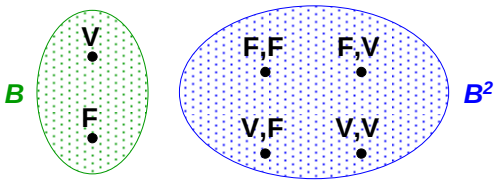
**Variabile dipendente:**
**B** (**"*andrò a fare una gita*"**)

**Relazione:**
**A → B** (**"*se A, allora B*"**)

## L'insieme B

**E' l'insieme dei valori Booleani:**

**B = { V, F }**




© A.Lioy - Politecnico di Torino (1992-2008)

## Operatori Booleani

- **operatori unari (es. Not )**
  **op : B → B**
- **operatori binari (es. And )**
  **op : $B^2$ → B**
- **descritti tramite una *tavola della verità***

## Tavola della verità (truth table)

| A | B | A *op* B |
|---|---|---|
| falso | falso | falso |
| falso | vero | falso |
| vero | falso | vero |
| vero | vero | falso |

## Espressioni Booleane

**Un'espressione Booleana è una combinazione di variabili ed operatori Booleani.**

**Ad esempio:**

**A e ( non B )**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Funzioni Booleane

**Una funzione Booleana è un'applicazione *molti-a-uno*:**

$$f : B^N \rightarrow B$$

**Ad esempio:**

**$f$ (A, B) = A e ( non B )**

## Operatore NOT

| A | $\overline{A}$ |
|---|---|
| falso | vero |
| vero | falso |

**Nota: per comodità grafica talvolta la negazione è indicata con un apice dopo la variabile o l'espressione negata (es. A')**

## Operatore AND

| A | B | A · B |
|---|---|---|
| falso | falso | falso |
| falso | vero | falso |
| vero | falso | falso |
| vero | vero | vero |

© A.Lioy - Politecnico di Torino (1992-2008)

## Operatore OR

| A | B | A + B |
|---|---|---|
| falso | falso | falso |
| falso | vero | vero |
| vero | falso | vero |
| vero | vero | vero |

## Operatore XOR

| A | B | A ⊕ B |
|---|---|---|
| falso | falso | falso |
| falso | vero | vero |
| vero | falso | vero |
| vero | vero | falso |

## Proprietà commutativa e associativa

**A · B = B · A**
**A + B = B + A**

**A · B · C = ( A · B) · C = A · (B · C) = (A · C) · B**
**A + B + C = (A + B) + C = A + (B + C) = (A + C) + B**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Proprietà distributiva

$A \cdot ( B + C ) = A \cdot B + A \cdot C$

$A + ( B \cdot C ) = ( A + B ) \cdot ( A + C )$

## Ordine delle operazioni

- **ordine degli operatori base:**

  **(1) NOT**
  **(2) AND**
  **(3) OR**

- **l'ordine delle operazioni può essere alterato mediante l'uso di parentesi (solitamente si usano solo parentesi tonde)**

## Dimostrazioni in algebra Booleana

**Siccome l'algebra Booleana contempla solo due valori è possibile effettuare le dimostrazioni (di proprietà o teoremi) considerando *esaustivamente* tutti i casi possibili:**

- **2 variabili → 4 combinazioni**
- **3 variabili → 8 combinazioni**
- **4 variabili → 16 combinazioni**
- **ecc.**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Dimostrazioni: un esempio

**A + ( B · C ) = ( A + B ) · ( A + C ) ?**

| A B C | A + B · C | (A + B) · (A + C) |
|---|---|---|
| 0 0 0 | 0 + 0 · 0 = 0 | (0+0) · (0+0) = 0 |
| 0 0 1 | 0 + 0 · 1 = 0 | (0+0) · (0+1) = 0 |
| 0 1 0 | 0 + 1 · 0 = 0 | (0+1) · (0+0) = 0 |
| 0 1 1 | 0 + 1 · 1 = 1 | (0+1) · (0+1) = 1 |
| 1 0 0 | 1 + 0 · 0 = 1 | (1+0) · (1+0) = 1 |
| 1 0 1 | 1 + 0 · 1 = 1 | (1+0) · (1+1) = 1 |
| 1 1 0 | 1 + 1 · 0 = 1 | (1+1) · (1+0) = 1 |
| 1 1 1 | 1 + 1 · 1 = 1 | (1+1) · (1+1) = 1 |

## Teoremi base

**A · A' = falso**
**A + A · B = A**
**A + A' · B = A + B**

**A + A' = vero**
**A · ( A + B ) = A**
**A · ( A' + B ) = A · B**

## Teorema di De Morgan

**Teorema:**

***f*** **( a, b, ..., z; +, · ) = *f '*( a', b', ..., z'; · , + )**

**ovvero (negando entrambi i membri):**

***f '*** **( a, b, ..., z; +, · ) = *f*** **( a', b', ..., z'; · , + )**

**Ad esempio:**

**A + B = ( A' · B' )'**
**( A + B )' = A' · B'**

© A.Lioy - Politecnico di Torino (1992-2008)

## Applicare il teorema di De Morgan

- **utile in programmazione per ridurre la complessità (e quindi il tempo di calcolo) di un'espressione Booleana**
- **i calcolatori impiegano lo stesso tempo a fare un test di uguaglianza o disuguaglianza**
- **esempio:**
  - **apertura porta con tesserino per lab in cui gli studenti sono esclusi**
  - **apri la porta se non è uno studente Polito o Erasmus**

```
if not
  ((id=s_polito) or (id=s_erasmus))
then open_door
```

4 operazioni

```
if (not (id=s_polito))
    and
   (not (id=s_erasmus))
then open_door
```

5 operazioni

```
if (id <> s_polito)
    and
   (id <> s_erasmus))
then open_door
```

3 operazioni

## Un aumento di capitale (problema)

**La società Alpha effettua un aumento di capitale gratuito: verranno assegnate 100 azioni ordinarie a chi possiede almeno 1000 azioni di risparmio, oppure a chi non le possiede ma ha sottoscritto un precedente aumento di capitale a pagamento.**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Un aumento di capitale (analisi)

**La società Alpha effettua un aumento di capitale gratuito: verranno assegnate 100 azioni ordinarie a chi possiede almeno 1000 azioni di risparmio, oppure a chi non le possiede ma ha sottoscritto un precedente aumento di capitale a pagamento.**

## Un aumento di capitale (variabili)

**Variabili indipendenti:**

**R (possesso di almeno 1000 azioni di risparmio)**

**P (sottoscrizione dell'aumento a pagamento)**

**Variabile dipendente:**

**A (assegnazione gratuita di 100 azioni ordinarie)**

## Un aumento di capitale (soluzione)

**Forma immediata:**

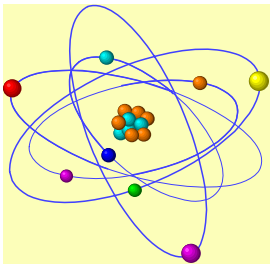
$$A = R + R' \cdot P$$

**Forma minima:**

$$A = R + P$$

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Tecnologia elettronica



## Elettrico o elettronico?

| ***elettrico*** | | ***elettronico*** |
|---|---|---|
| **tanti** | **[elettroni ]** | **pochi** |
| **O( 100V )** | **[ tensione ]** | **O( V )** |
| **O( A )** | **[ corrente ]** | **O( mA )** |

## Analogico o digitale?

| ***analogico*** | ***digitale*** |
|---|---|
| **proporzionale** | **numerico** |

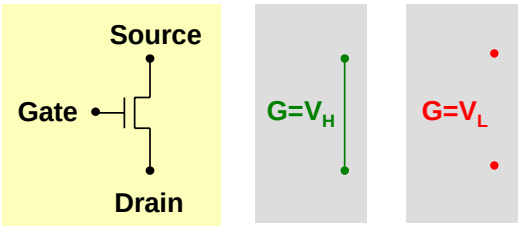
*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il transistore

**È l'elemento base degli attuali elaboratori *elettronici digitali*:**

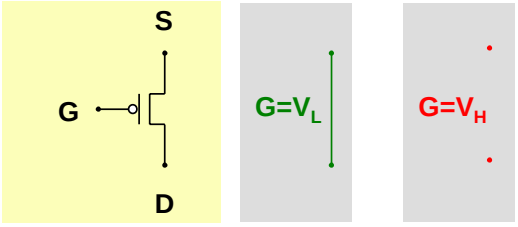
- **piccole dimensioni ( nm )**
- **basso consumo di energia ( μW )**
- **alta velocità di funzionamento (milioni di operazioni al secondo)**

## Il transistore MOS (Metal-Oxide Semiconductor)




**( transistore MOS a canale *n* , o *nMOS* )**

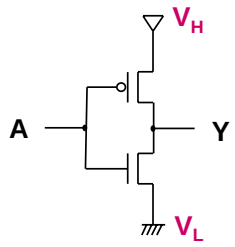
## Il transistore MOS (Metal-Oxide Semiconductor)




**( transistore MOS a canale p , o pMOS)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il circuito Inverter

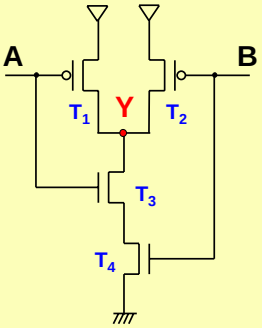



$A = V_H \rightarrow Y = V_L$
$A = V_L \rightarrow Y = V_H$

## Analisi di un circuito a transistori MOS

- dato un insieme di tensioni sugli ingressi
- si esaminano le tensioni di gate per stabilire lo stato dei transistori (conduce, non conduce)
- partendo dall'uscita del circuito si esaminano i percorsi che conducono verso una tensione definita (VH o VL)
- se tutti i cammini portano ad un'unica tensione l'uscita assumerà lo stesso valore, altrimenti il suo valore sarà indefinito

## Esempio: analisi di un circuito MOS




| A | B | $T_1$ | $T_2$ | $T_3$ | $T_4$ | Y |
|---|---|---|---|---|---|---|
| *L* | *L* | *on* | *on* | *off* | *off* | *H* |
| *L* | *H* | *on* | *off* | *off* | *on* | *H* |
| *H* | *L* | *off* | *on* | *on* | *off* | *H* |
| *H* | *H* | *off* | *off* | *on* | *on* | *L* |

**( porta NAND )**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Le porte logiche

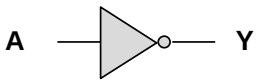
**Associando a $V_H$ e $V_L$ due valori logici distinti è possibile *interpretare* il funzionamento dei circuiti elettronici a transistori come operatori Booleani.**

**Associando invece due cifre binarie distinte è possibile *interpretare* il funzionamento in modo aritmetico.**

## Trivalenza di una porta logica

| *A* | *Y* | *interpretazione* |
|---|---|---|
| $V_L$ | $V_H$ | circuito Inverter |
| $V_H$ | $V_L$ | |
| falso | vero | operatore Booleano NOT |
| vero | falso | |
| 0 | 1 | operazione binaria |
| 1 | 0 | complemento a uno |

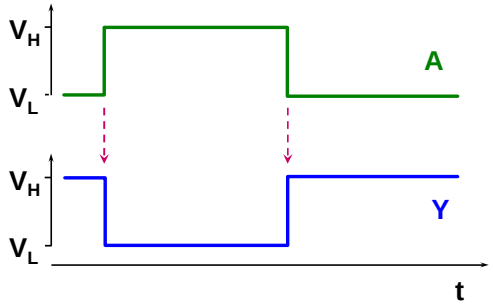
## La porta INV (Inverter)



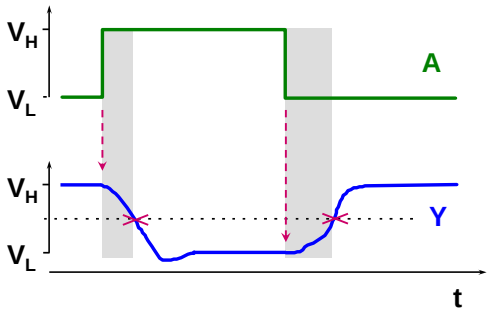

**Y = A'**

© *A.Lioy - Politecnico di Torino (1992-2008)*

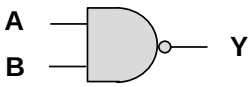
## INV: funzionamento ideale




## INV: funzionamento reale



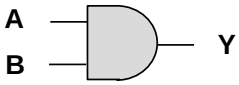

## La porta NAND




$Y = ( A \cdot B )' = A' + B'$

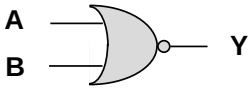
*© A.Lioy - Politecnico di Torino (1992-2008)*

## La porta AND



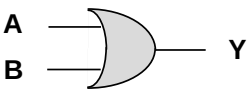

$Y = A \cdot B$

## La porta NOR




$Y = ( A + B )' = A' \cdot B'$

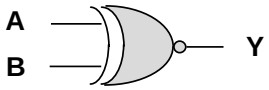
## La porta OR




$Y = A + B$

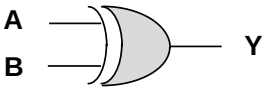
*© A.Lioy - Politecnico di Torino (1992-2008)*

## La porta XNOR (o EQV)

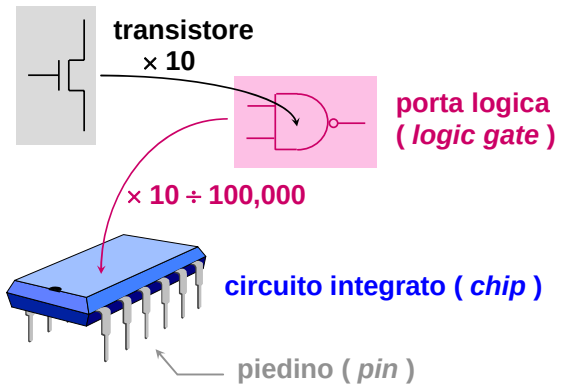



$Y = ( A \oplus B )' = A \cdot B + A' \cdot B'$

## La porta XOR

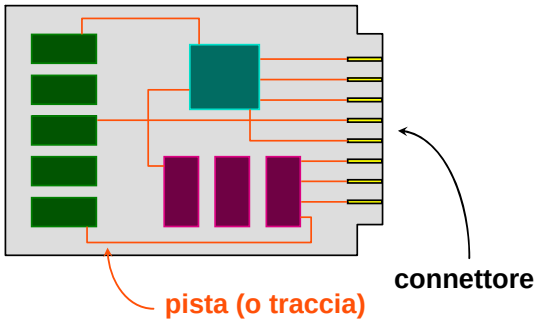



$Y = A \oplus B = A \cdot B' + A' \cdot B$




© A.Lioy - Politecnico di Torino (1992-2008)

## Piastra elettronica ( *board* )




## Parametri tecnologici

- densità di integrazione
- velocità
  → tempo di propagazione
- potenza elettrica
  → assorbimento di energia
  → sviluppo di calore
- capacità di pilotaggio

## Densità di integrazione

**SSI** : ***Small Scale of Integration***
**MSI** : ***Medium Scale of Integration***
**LSI** : ***Large Scale of Integration***
**VLSI** : ***Very Large Scale of Integration***
**ULSI** : ***Ultra Large Scale of Integration***

© A.Lioy - Politecnico di Torino (1992-2008)

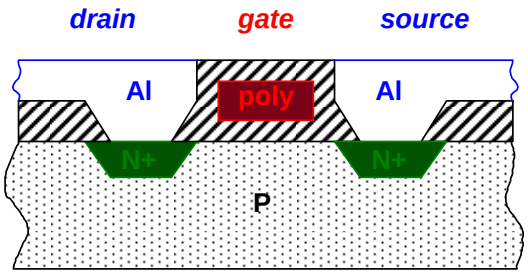
## Tecnologie a confronto

**Tecnologie del silicio:**

- **TTL (*Transistor-Transistor Logic*)**
- **CMOS (*Complementary Metal-Oxide Semiconductor*)**
- **ECL (*Emitter-Coupled Logic*)**
- **BiCMOS ( = TTL + CMOS )**

**Tecnologia dell'arseniuro di gallio (GaAs)**

## Schema di un transistore nMOS




## Circuiti logici combinatori

**Sono circuiti il cui valore delle uscite dipende esclusivamente dai valori presenti sugli ingressi ad un *singolo istante* di tempo.**

**In altre parole, il valore di uscita è una *combinazione* dei valori presenti in ingresso ad un determinato istante.**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Circuiti combinatori

**Circuiti combinatori ideali:**

$$U_t = f ( X_t )$$

**Circuiti combinatori reali:**

$$U_t = f ( X_{t - \Delta} )$$

## Velocità e ritardo

**$\Delta$ è detto *ritardo di propagazione* ed è il tempo che i segnali impiegano a propagarsi all'interno del circuito.**

**La *velocità di funzionamento* (massima) è:**

$$v = 1 / \Delta$$

## Velocità e ritardo

**Esempio - porta And:**

- **ritardo = 4 ns**
- **velocità = 250 milioni AND/s**

**Esempio - sommatore:**

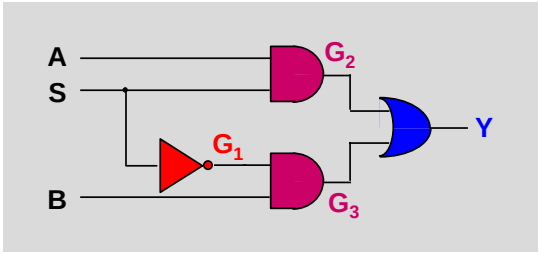
- **ritardo = 20 ns**
- **velocità = 50 milioni somme/s**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Analisi di un circuito combinatorio

- **per ogni possibile configurazione di valori logici sugli ingressi ...**
- **... si calcola il valore delle uscite delle porte che hanno tutti gli ingressi specificati ...**
- **... fino ad arrivare alle uscite del circuito**

## Multiplexer a 1 bit, 2 vie




## Analisi del MUX 1:2

| A B S | $G_1$ | $G_2$ | $G_3$ | Y |
|---|---|---|---|---|
| 0 0 0 | 1 | 0 | 0 | 0 |
| 0 0 1 | 0 | 0 | 0 | 0 |
| 0 1 0 | 1 | 0 | 1 | 1 |
| 0 1 1 | 0 | 0 | 0 | 0 |
| 1 0 0 | 1 | 0 | 0 | 0 |
| 1 0 1 | 0 | 1 | 0 | 1 |
| 1 1 0 | 1 | 0 | 1 | 1 |
| 1 1 1 | 0 | 1 | 0 | 1 |

**S=1 → Y=A** **S=0 → Y=B**

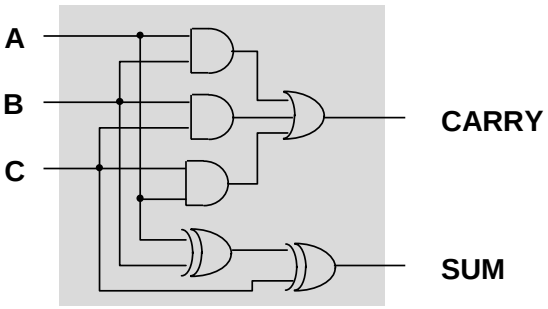
*© A.Lioy - Politecnico di Torino (1992-2008)*

## MUX: logica

$Y = G_2 + G_3$
$= (A \cdot S) + (G_1 \cdot B)$
$= A \cdot S + S' \cdot B$

( S=0 ) Y = A · 0 + 1 · B = 0 + B = B
( S=1 ) Y = A · 1 + 0 · B = A + 0 = A

## Sommatore completo ( *full-adder* )




## Full-adder: logica

SUM
$= A \oplus B \oplus C$
$= A \cdot B' \cdot C' + A' \cdot B \cdot C' + A' \cdot B' \cdot C$
$+ A \cdot B \cdot C$

CARRY
$= A \cdot B + A \cdot C + B \cdot C$

© A.Lioy - Politecnico di Torino (1992-2008)

## Full-adder: tavola della verità

| A B C | Sum | Carry |
|---|---|---|
| 0 0 0 | 0 | 0 |
| 0 0 1 | 1 | 0 |
| 0 1 0 | 1 | 0 |
| 0 1 1 | 0 | 1 |
| 1 0 0 | 1 | 0 |
| 1 0 1 | 0 | 1 |
| 1 1 0 | 0 | 1 |
| 1 1 1 | 1 | 1 |

## Circuiti sequenziali

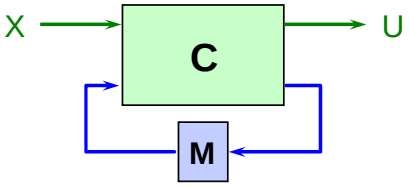
**Sono circuiti il cui valore delle uscite dipende dai valori presenti sugli ingressi a *diversi istanti* di tempo.**

**In altre parole, il valore di uscita dipende dalla *sequenza* dei valori in ingresso.**

**Anche detti circuiti con *memoria*.**

## Circuiti sequenziali

$U_t = f ( X_{t'} , X_{t''} , \dots ) = g (X_{t'} , M_{t'} )$




© *A.Lioy - Politecnico di Torino (1992-2008)*

## Sincrono o asincrono?

**Funzionamento *sincrono*:**

- **esiste un segnale di riferimento (solitamente detto *clock* ) che sincronizza le operazioni di tutti i circuiti**
- **ordine → facilità di progetto**
- **si attende il circuito più lento!**

## Sincrono o asincrono?

**Funzionamento *asincrono*:**

- **ogni circuito esegue il proprio lavoro in modo indipendente dagli altri ... non appena i dati sono pronti ... o anche quando solo una parte dei dati è disponibile!**
- **grande velocità**
- **disordine → difficoltà di progetto**

## Sincrono o asincrono?

**Gli elaboratori elettronici digitali funzionano al 99% in modo sincrono perché, contenendo milioni di elementi logici, è praticamente impossibile progettarli diversamente.**

**In futuro possibili incrementi di prestazioni deriveranno anche dal passaggio al modo asincrono.**

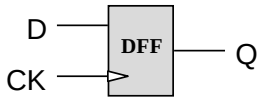
*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il flip-flop

**È l'elemento base dei circuiti sequenziali:**

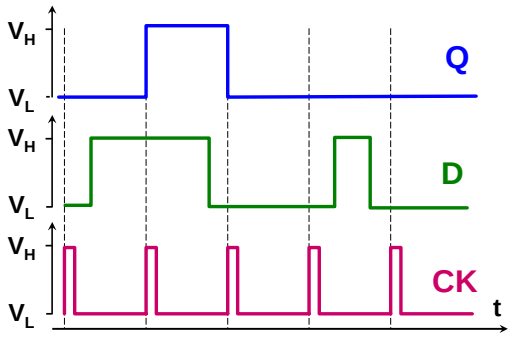
- **memorizza un bit**
- **funzionamento sincrono o asincrono**

## Il flip-flop di tipo D sincrono




| D | CK | Q |
|---|---|---|
| 0 | ↑ | 0 |
| 1 | ↑ | 1 |
| 0,1 | 0,1 | $Q_{-1}$ |

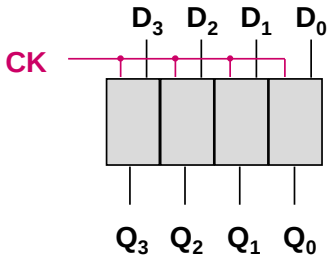
## Il flip-flop di tipo D




© A.Lioy - Politecnico di Torino (1992-2008)

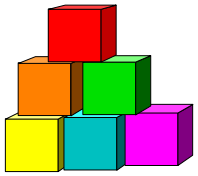
## I registri

**Un registro è un insieme di flip-flop comandati da un unico clock.**

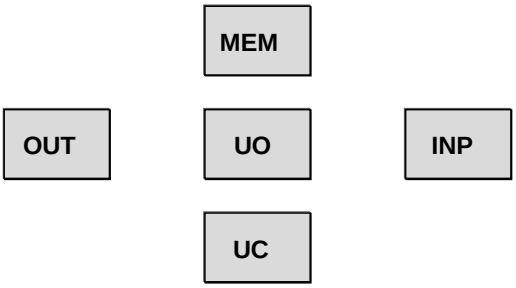



*© A.Lioy - Politecnico di Torino (1992-2008)*

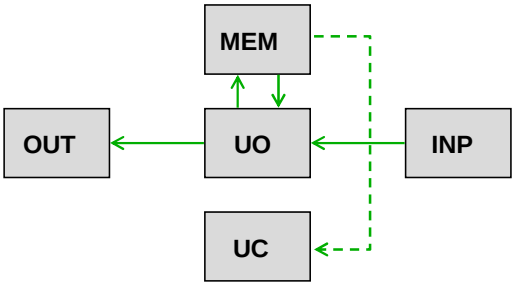
## Architettura di un elaboratore elettronico



## Blocchi funzionali

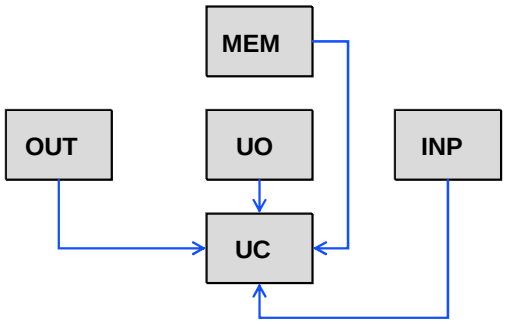



## Flusso dei dati

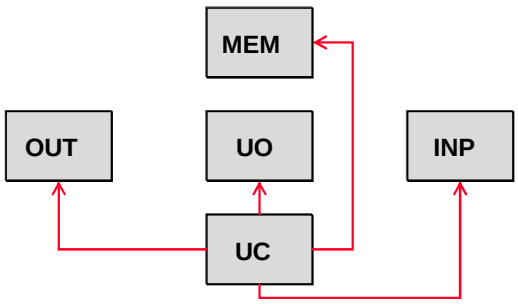



© A.Lioy - Politecnico di Torino (1992-2008)

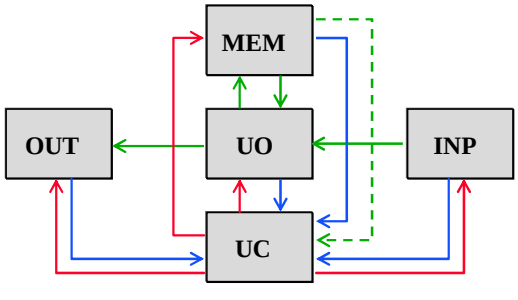
## Informazioni di stato




## Ordini

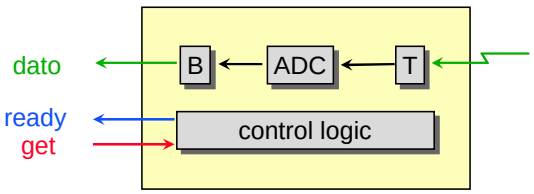



## Struttura a blocchi




*© A.Lioy - Politecnico di Torino (1992-2008)*

## Unità di input: schema funzionale




## Unità di input

**Acquisisce informazioni trasformandole dal mondo umano a quello del computer:**

- **umano = diversi tipi di segnali fisici, analogici, asincroni**
- **computer = segnali elettronici, digitali, sincroni**

## Trasduttore

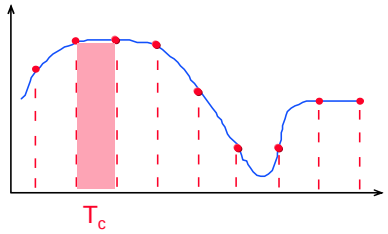
- **trasforma un segnale da un dominio fisico ad un altro**
- **nei calcolatori, tipicamente si usano trasduttori elettro-xxx**
- **esempi:**
  - **elettro-ottico (cellula fotoelettrica)**
  - **elettromeccanico (tastiera)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Analog-Digital-Converter

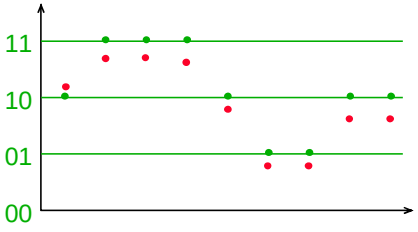
- **trasforma un segnale elettrico analogico**
- **... in un segnale elettronico digitale**
- **risoluzione temporale (velocità)**
- **risoluzione di valori (n. di bit)**

## Campionamento




**frequenza di campionamento:** $f_C = 1/T_C$

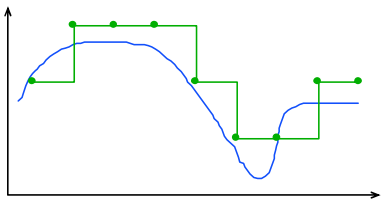
## Quantizzazione




**K livelli** $\rightarrow$ $n = \lceil \log_2 K \rceil$ **bit**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Errori di campionamento e di quantizzazione



## Problema

***Un sistema di elaborazione deve acquisire una temperatura compresa tra 35º e 43ºC con la precisione di 0.1ºC.***

***Qual è il numero minimo di bit di cui deve disporre l'ADC?***

## Soluzione

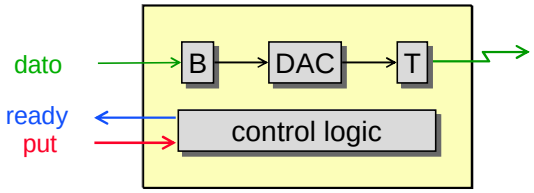
- **intervallo:** **[ 35.0 ... 43.0 ]**
- **valori distinti:** **81**
- **n. minimo di bit:** **7**
- ***qual è la risoluzione ottenuta?***

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Buffer

- **elemento di memoria locale**
- **conserva i dati nell'attesa che vengano prelevati**
- **permette di continuare ad acquisire nuovi dati (equalizzazione delle velocità)**

## Unità di output: schema funzionale




## Unità operativa

**Svolge tutte le elaborazioni richieste (aritmetiche, logiche, grafiche, ...).**

**E' composta di:**

- **ALU**
- **flag**
- **registri**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## ALU (Arithmetic-Logic Unit)

- **svolge tutti i calcoli (aritmetici e logici)**
- **solitamente composta da circuiti combinatori**

## Flag

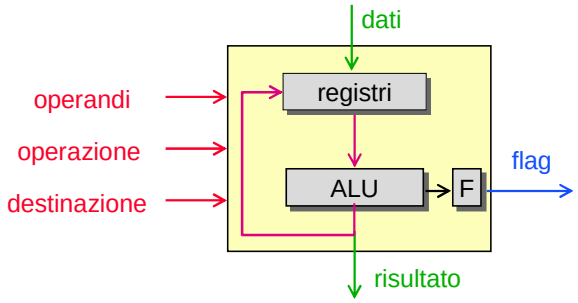
- **indicatori di stato del risultato dell'operazione della ALU**
- **sono singoli bit (0=falso, 1=vero)**
- **solitamente raggruppati in un registro**
- **flag più comuni:**
  **Z (zero)** **V (overflow)**
  **CY (carry)** **N (negative)**

## Registri

- **elementi di memoria usati per conservare *temporaneamente* dei dati (es. risultati parziali).**
- **pochi (8...128)**
- **dimensione di una *word* (8...64 bit)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Unità operativa: schema funzionale




## CPU e FPU

**Central Processing Unit (CPU):**

- **CPU = UO + UC**
- **microprocessore (µP) = CPU + "frattaglie"**

**Floating Point Unit (FPU):**

- **UO dedicata ai numeri reali**
- **alias "coprocessore matematico"**

## Memoria

**Memorizza i dati e le istruzioni necessarie all'elaboratore per operare.**

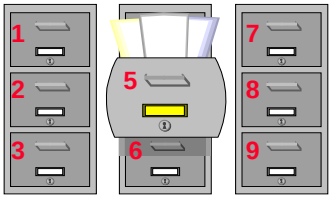
**Caratteristiche:**

- **indirizzamento**
- **parallelismo**
- **accesso (sequenziale o casuale)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Indirizzamento

**Ad ogni cella di memoria è associato un *indirizzo* (numerico) per identificarla univocamente.**




## Parallelismo

**Ogni cella di memoria contiene una quantità fissa di bit:**

- **identica per tutte le celle (di una certa unità di memoria)**
- **accessibile con un'unica istruzione**
- **è un multiplo del byte**
- **minimo un byte**

## Memoria interna

- **all'interno dell'elaboratore**
- **è allo stato solido (*chip*)**
- **solitamente è volatile**
- **veloce (nanosecondi, $10^{-9}$s)**
- **quantità limitata (qualche GB)**
- **non rimovibile**
- **costosa (0,5 - 1.0 € / MB)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Memoria esterna

- **all'esterno dell'elaboratore**
- **talvolta rimovibile**
- **non elettronica (es. magnetica)**
- **permanente**
- **lenta (millisecondi, $10^{-3}$ s)**
- **grande quantità (qualche GB)**
- **economica (1 - 2 € / GB)**

## Memoria ad accesso casuale

- **il tempo di accesso è costante (indipendente dalla cella scelta)**

  $T_a$ = costante

- **anche detta RAM (Random Access Memory)**

## Memoria ad accesso sequenziale

- **il tempo di accesso dipende dalla cella a cui si accede**
- **spesso la dipendenza è lineare**

  $T_a = K \cdot$ *indirizzo*

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Memoria RAM (Random Access Memory)

- **circuiti integrati**
- **memoria *statica ( SRAM )***
- **memoria *dinamica ( DRAM )***
  **(richiede un *rinfresco* periodico)**
- **ormai sinonimo di memoria interna volatile casuale a lettura e scrittura**

## Memoria ROM (Read-Only Memory)

**E' un concetto (memorie a sola lettura ) … ma anche una classe di dispositivi allo stato solido (memorie a prevalente lettura = molto più veloce o facile della scrittura).**

**ROM**

- **dati scritti in fabbrica**

**PROM (Programmable ROM)**

- **dati scritti dall'utente tramite un apparecchio speciale (programmatore)**

**EPROM (Erasable PROM)**

- **PROM cancellabile tramite UV**

**EAROM (Electrically Alterable ROM)**

- **PROM cancellabile tramite circuito elettronico speciale**

**EEPROM, $E^2$PROM (Electrically Erasable PROM)**

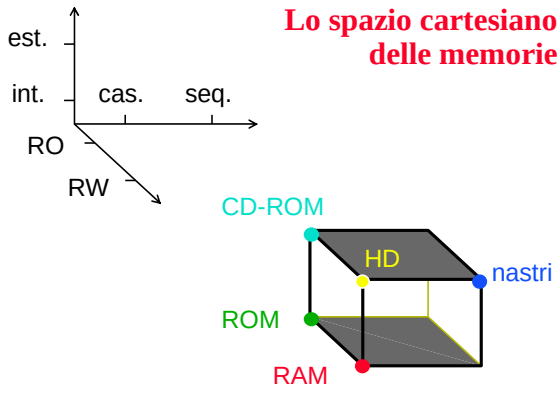
- **scrivibile/cancellabile normalmente**

**Flash memory**

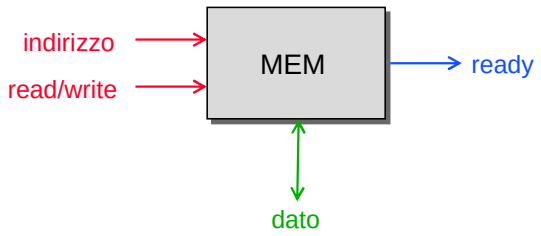
- **EEPROM veloce nella cancellazione (un blocco/tutta invece di un byte alla volta)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

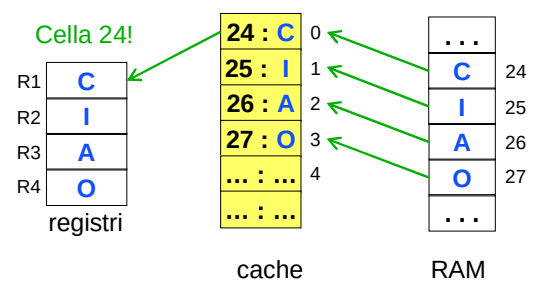
## Lo spazio cartesiano delle memorie




## Memoria: schema funzionale

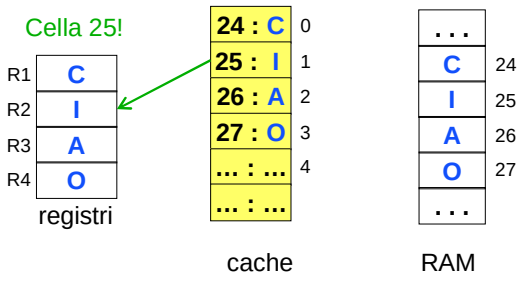



## La memoria cache (caricamento)




© A.Lioy - Politecnico di Torino (1992-2008)

## La memoria cache (uso)




## Caratteristiche della cache

- **copia in un colpo solo un intero *blocco di memoria* dalla RAM**
- **ha un tempo di accesso più vicino a quello dei registri che a quello della RAM**
- **è indirizzabile in base al contenuto (CAM, Content Addressable Memory):**
  - **ad esempio: "voglio la cella 25!"**
  - **altrimenti non potrebbe funzionare perché il programma non sa in quale cella della cache si trovi il dato desiderato**

## Prestazioni della cache

**Tempo medio di accesso in memoria:**

$$T_M = H \cdot T_{cache} + ( 1 - H ) \cdot T_{RAM}$$

**ove H = *hit ratio* (percentuale di celle trovate nella cache rispetto al totale degli accessi in memoria)**

**$H \geq 90\% \rightarrow T_M \cong T_{cache}$**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Prestazioni della cache

**$T_{cache}$ = 20 ns**
**$T_{RAM}$ = 80 ns**
**H = 60%**
**$T_M = 0.6 \cdot 20 + 0.4 \cdot 80 = 44$ ns**
**H = 95%**
**$T_M = 0.95 \cdot 20 + 0.05 \cdot 80 = 23$ ns**

## Livelli di cache

- **nelle CPU ad alta velocità, la differenza di velocità tra RAM e registri è molto elevata e quindi si spezza la cache in vari livelli: L1, L2, talvolta anche L3**
- **tipicamente almeno L1 cache è on-chip, ossia integrata sul medesimo chip della CPU**
- **cache dedicata ai dati (d-cache)**
- **cache dedicata alle istruzioni (i-cache)**

## La cache dei dischi

**Essendo i dischi abissalmente lenti rispetto alla memoria centrale, è possibile usare quest'ultima come cache per gli ultimi dati letti:**

- **disk cache**
- **read-ahead**

**Sempre più spesso la disk cache è integrata all'interno del controller del disco.**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Gerarchia di memoria

## Unità di controllo

**E' il cuore dell'elaboratore:**

- **in base al programma fornitole ...**
- **ed allo stato di tutte le unità ...**
- **decide l'operazione da eseguire ...**
- **ed emette gli ordini relativi**

## I programmi

**Un *programma* è una specifica univoca di una serie di operazioni che l'elaboratore deve svolgere:**

- **costituito da una sequenza di *istruzioni***
- **specificato in *codice macchina***
- **conservato in *memoria***

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Le istruzioni

**Ogni istruzione specifica un'operazione elementare che l'elaboratore deve svolgere:**

- ***funzione*** **(es. ADD, MOV )**
- ***operandi*** **(es. R1, M[126] )**
- ***destinazione*** **del risultato (stessa sintassi degli operandi)**

## Il codice macchina

**E' un particolare codice binario usato per specificare le istruzioni all'elaboratore:**

- **diverso per ogni elaboratore (perché ogni elaboratore ha un'organizzazione di MEM e UO diversa)**
- **formato fisso (= semplicità)**
- **formato variabile (= efficienza)**

## Un esempio di codice macchina

**Istruzioni codificate su 16 bit:**

- **operazioni su 4 bit → 16 operazioni**
- **operandi su 4 bit → 16 registri**
- **indirizzi su 8 bit → max 256 celle**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Un esempio di codice macchina

**Operazioni tra registri:**

| 4 | 4 | 4 | 4 |
|---|---|---|---|
| *operazione* | *dest* | *op1* | *op2* |
| Add | Rz | Rx | Ry |
| Mov | Rz | Rx | - |
| Not | Rz | - | - |

## Un esempio di codice macchina

**Operazioni su memoria, periferici o dati immediati:**

| 4 | 4 | 8 |
|---|---|---|
| *operazione* | *reg* | *costante* |
| Load | Rz | indirizzo |
| Inp | Rz | indirizzo |
| Mvi | Rz | dato |

## Codici delle operazioni

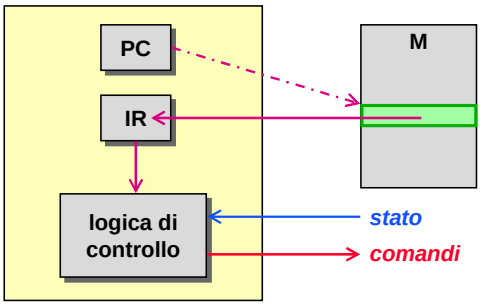
| | | | |
|---|---|---|---|
| Nop | 0000 | Load | 1000 |
| Add | 0001 | Store | 1001 |
| Sub | 0010 | Inp | 1010 |
| And | 0011 | Out | 1011 |
| Or | 0100 | Jmp | 1100 |
| Not | 0101 | Jp | 1101 |
| Mov | 0110 | Jz | 1110 |
| Mvi | 0111 | Stop | 1111 |

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Un esempio di codice macchina

| *dati in memoria* | *interpretazione* |
|---|---|
| 1010 0000 00000101 | R0 ← perif(5) |
| 1010 0001 00000101 | R1 ← perif(5) |
| 0001 0010 0000 0001 | R2 ← R0 + R1 |
| 1011 0010 00000110 | perif(6) ← R2 |

## Unità di controllo: schema funzionale




## Componenti dell'UC

- **PC (Program Counter)**
  **registro che indica sempre l'indirizzo della cella di memoria che contiene la *prossima* istruzione da eseguire**
- **IR (Instruction Register)**
  **registro che memorizza *temporaneamente* l'operazione corrente da eseguire**
- **logica di controllo**
  **interpreta il codice macchina in IR per decidere ed emette gli ordini che le varie unità devono eseguire**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Esecuzione di un'istruzione

**Tre fasi distinte:**

- ***fetch*** **IR ← M [ PC ]**
  **PC ← PC + 1**
- ***decode*** **ordini ← decode(IR)**
- ***execute*** **ready? go!**

## Inizializzazione

**Cosa succede all'accensione di un calcolatore?**

- **nel PC viene forzato l'indirizzo della cella di memoria ove inizia il primo programma da eseguire (programma di *bootstrap*)**
- **questo programma iniziale risiede in ROM**

## Istruzioni di salto

**Non tutti i programmi sono una sequenza lineare di istruzioni.**

**Ad esempio (radice quadrata di N):**

- **se N ≥ 0, allora**

$$R = \sqrt{N}$$

- **altrimenti**

$$R = i \cdot \sqrt{-N}$$

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Istruzioni di salto

- **salti incondizionati (Jump)**

  **JMP 1204 ; PC ← 1204**

- **salti condizionati (Jump-if)**

  **SUB R1,R0,17 ; R1 ← R0 – 17**
  **JP 1204 ; se > 0, PC ← 1204**

## Istruzioni di salto: esempio

**Ipotesi: R1 contiene i ricavi, R2 le spese**

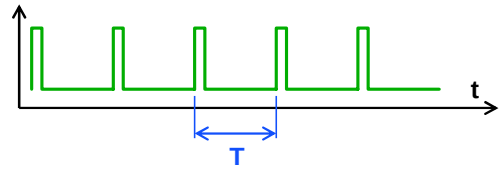
```
        SUB R0,R1,R2    ; R0 ← R1 - R2
        JP 1492         ; salto se R1>R2
        ... (chiedo il rimborso)
        JMP 1500
(1492)  ... (calcolo le tasse da pagare)
        ...
(1500)  STOP
```

## Il clock

**Ogni elaboratore contiene un elemento di temporizzazione (detto *clock* ) che genera un riferimento temporale comune per tutti gli elementi costituenti il sistema di elaborazione.**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il clock




- **T = *periodo* di clock**
  - **unità di misura = s**
- **f = *frequenza* di clock ( = 1 / T )**
  - **unità di misura = $s^{-1}$ = Hz (cicli/s)**

## Tempistica delle istruzioni

- **un *ciclo-macchina* è l'intervallo di tempo in cui viene svolta una operazione elementare ed è un multiplo del periodo del clock**
- **l'esecuzione di un'istruzione richiede un numero intero di cicli macchina, variabile a seconda del tipo di istruzione**

## Velocità di elaborazione

**Valutazione molto difficile perché dipende pesantemente da:**

- **programma eseguito**
- **dati**
- **architettura dell'elaboratore (istruzioni macchina, cache, ...)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## MIPS
## ( Million Instructions Per Second )

**f = frequenza di clock [ Hz = cicli/s ]**
**T = periodo di clock = 1 / f [ s ]**
**C = cicli macchina / istruzione**
**IPS = f / C = 1 / ( T × C )**
**MIPS = IPS / $10^6$**

## MIPS

**Definizione alternativa:**

**M**eaningless
**I**nformation
**P**rovided by
**S**alesman

**Probabilmente è molto più realistica di quella ufficiale !**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## MFLOPS ( Million FLoating-point Operations Per Second )

- **velocità di elaborazione per problemi di tipo scientifico**
- **dubbi analoghi a quelli per i MIPS**

## Benchmark

**Drysthone**

- **numeri interi**

**Linpack**

- **algebra lineare (interi)**

**SPECmark**

- **media geometrica di più prove**
- **SPECint2000 – interi**
  **12 prove (11 C, 1 C++)**
- **SPECfp2000 – floating point**
  **14 prove (4 C, 6 Fortran77, 4 Fortran90)**

**SPECint2000**

| | | |
|---|---|---|
| gzip | C | compressione |
| vpr | C | FPGA placement and routing |
| gcc | C | compilatore C |
| mcf | C | ottimizzazione combinatoria |
| crafty | C | gioco degli scacchi |
| parser | C | elaborazione testi |
| eon | C++ | grafica (visualizzazione) |
| perlbmk | C | linguaggio PERL |
| gap | C | interprete (teoria dei gruppi) |
| vortex | C | database object-oriented |
| bzip2 | C | compressione |
| twolf | C | emulatore di placement e routing |

**SPECfp2000**

| | | |
|---|---|---|
| wupwise | F77 | fisica (cromodinamica quantistica) |
| swim | F77 | modello acque poco profonde |

*© A.Lioy - Politecnico di Torino (1992-2008)*

| | | |
|---|---|---|
| mgrid | F77 | multi-grid solver: 3D potential field |
| applu | F77 | equazioni alle derivate parziali |
| mesa | C | libreria grafica 3-D |
| galgel | F90 | fluidodinamica computazionale |
| art | C | reti neurali (riconoscimento immagine) |
| equake | C | simulazione propagazione onde sismiche |
| facerec | F90 | elaborazione immagini (riconoscimento di un volto) |
| ammp | C | chimica computazionale |
| lucas | F90 | test di primalità |
| fma3d | F90 | simulatore di crash (tramite il metodo degli elementi finiti) |
| sixtrack | F77 | progetto di un'acceleratore nucleare per alte energie |
| apsi | F77 | meteorologia (diffusione inquinanti) |

## Valutazione di un sistema di elaborazione

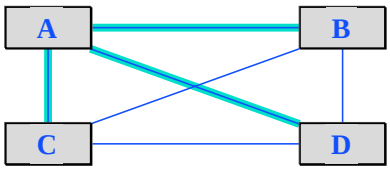
**Altri criteri in dipendenza dal tipo di applicazione:**

- **velocità di I/O**
- **numero di utenti**
- **numero di transazioni (TPS)**

## Struttura a bus

***Problema*: collegamento completo tra N unità richiede N×(N-1)/2 collegamenti**

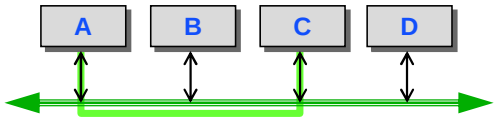
***Esempi*: 4U:6C, 10U:45C, 20U:190C**




© A.Lioy - Politecnico di Torino (1992-2008)

## Struttura a bus

***Soluzione*:**
**segnali omologhi raggruppati in un *bus*, usato a turno dalle varie unità**

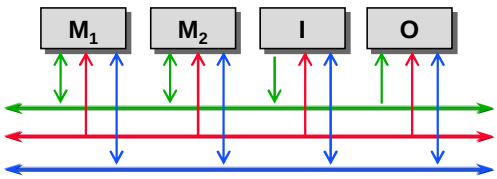



## Caratteristiche di un bus

- **trasporto di un solo dato per volta**
- **frequenza = n. di dati trasportati al secondo**
- **ampiezza = n. di bit di cui è costituito un singolo dato**
- **se mal dimensionato, potrebbe essere un collo di bottiglia**

## Tipi fondamentali di bus

- **bus dati ( DBus )**
- **bus degli indirizzi ( ABus )**
- **bus di controllo ( CBus )**




*© A.Lioy - Politecnico di Torino (1992-2008)*

## CPU Intel per PC

| CPU | DBus | ABus | Cache | FPU |
|---|---|---|---|---|
| 8088 | 8 bit | 20 bit | No | No |
| 8086 | 16 bit | 20 bit | No | No |
| 80286 | 16 bit | 24 bit | No | No |
| 80386 | 32 bit | 32 bit | No | No |
| 80486 | 32 bit | 32 bit | 8 KB | Sì |
| Pentium | 64 bit | 32 bit | 8+8KB | Sì |
| P-PRO | 64 bit | 32 bit | 8+8/256 | Sì |

## Massima memoria interna (fisicamente presente)

- **la dimensione dell'Abus determina il max numero di celle di memoria**
- **la dimensione del Dbus "suggerisce" la dimensione di una cella di memoria (ma sono possibili anche celle di maggior dimensione che richiedono due o più trasferimenti sul Dbus)**
- **max mem = $2^{|Abus|}$ x |Dbus|**
- **esempio (Abus da 20 bit, Dbus da 16 bit):**
  **max mem = $2^{20}$ x 2 byte = 2 MB**
  **ossia 1 M celle di memoria, ognuna da 2 byte**

## Massima memoria esterna

- **la memoria esterna non dipende dall'Abus perché viene vista come un periferico (di input e/o di output)**
- **la massima quantità di memoria esterna dipende dal bus di I/O (quello su cui sono collegati i periferici)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Storia delle CPU Intel per PC

| CPU | anno | Dhry. MIPS | transistor |
|---|---|---|---|
| 8086 | giu-78 | 0.3 - 0.7 | 29,000 |
| 80286 | feb-82 | 1 - 3 | 134,000 |
| 80386 | ott-85 | 5 - 10 | 275,000 |
| 80486 | ago-89 | 20 - 50 | 1,200,000 |
| Pentium | mar-93 | 100 - 150 | 3,100,000 |
| P-PRO | ott-95 | 250+ | 5,500,000 |

## CPU Pentium

- **Pentium-II = PentiumPRO + MMX**
- **MMX =**
  - **16+16K di L1-cache**
  - **57 nuove istruzioni multimediali**
  - **istruzioni SIMD**
- **Pentium-III = Pentium-II +**
  **70 nuove istruzioni per grafica 3D**
  **e streaming audio/video**
- **Pentium 4 = nuova architettura**

## CPU Intel: modelli particolari

- **80386 SX**
  **(DBus=16 bit, ABus=24 bit)**
- **80486 SX (priva di FPU)**
- **80386/486 SL (SX + Low-power)**
- **80486 DX2 / DX4 (clock esterno pari a metà /**
  **un quarto di quello interno)**
- **P-II Celeron (senza L2 cache)**
- **P-III Celeron (128 KB L2 cache)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Bus standard di I/O

**Bus per Personal Computer:**

- **ISA (Industry Standard Architecture)**
- **EISA (Extended ISA)**
- **MCA (MicroChannel Architecture)**

**Bus per workstation/industria:**

- **VME**
- **Futurebus**

**Unificazione:**

- **PCI (Peripheral Connect Interface)**

## PCI

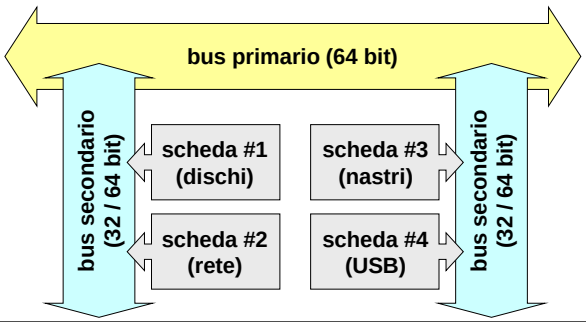
- **gerarchia di bus a due livelli**
- **livello base = 32 bit / 66 MHz, oppure 64 bit / 33 MHz**
- **livello avanzato = 64 bit / 66 MHz**
- **max troughput tra CPU e periferici = 532 MB/s**

## PCI-X

- **PCI eXtended (o PCI eXpress)**
- **gerarchia di bus a due livelli**
- **livello base= 64 bit / 66 MHz**
- **livello avanzato = 64 bit / 133 MHz**
- **max troughput tra CPU e periferici = 1.06 GB/s**
- **compatibile con schede PCI normali**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## PCI / PCI-X


bus primario (64 bit)
bus secondario (32 / 64 bit)
bus secondario (32 / 64 bit)
scheda #1 (dischi)
scheda #3 (nastri)
scheda #2 (rete)
scheda #4 (USB)

## Bus standard - confronto

| bus | DBus [ bit ] | ABus [ bit ] | velocità [ MB/s ] |
|---|---|---|---|
| ISA | 8, 16 | 20, 24 | 5.3 |
| EISA | 8, 16, 32 | 20, 24, 32 | 33 |
| MCA | 8,16, 32 | 20, 32 | 20 |
| **PCI** | **32, 64** | **32** | **133 / 532** |
| **PCI-X** | **64** | **32** | **532 / 1060** |

## Supercalcolatori

**Sistemi multiprocessori:**

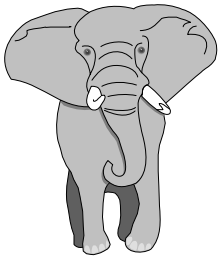
- **hanno un elevato numero di CPU**
- **SIMD (Single Instruction, Multiple Data); es. Connection Machine (max 65536 UO)**
- **MIMD (Multiple Instruction, Multiple Data); es. NCUBE (max 2048 CPU)**

**Sistemi vettoriali:**

- **hanno UO in grado di svolgere operazioni tra vettori; es. CRAY**

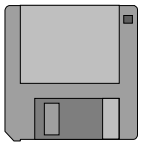
*© A.Lioy - Politecnico di Torino (1992-2008)*

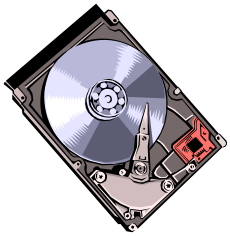
## Dispositivi di memoria di massa



## Categorie di dispositivi

**dischi**





**nastri**

## Valutazione della prestazioni

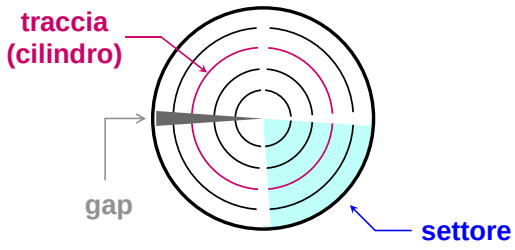
- **capacità:**
  - **massima quantità di dati memorizzabile**
- **tempo di accesso (seek time):**
  - **tempo per il posizionamento dell'elemento di lettura/scrittura su un blocco di dati**
- **velocità di trasferimento (throughput):**
  - **massima quantità di dati trasferibile nell'unità di tempo**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## I dischi

- **accesso casuale**
- **tempo di accesso: O (10 ms)**
- **magnetici, ottici, magneto-ottici**
- **dischi fissi o rimovibili**

## Organizzazione dei dischi

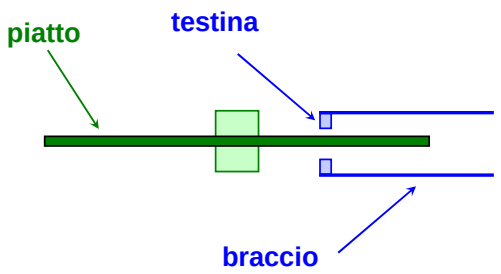



## Dischi magnetici - tecnologia

- **supporto rigido ricoperto di materiale ferromagnetico**
- **testina di lettura/registrazione (elettrocalamita)**
- **la testina poggia sul disco (se rimovibile) oppure lo sfiora a circa 1µm**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Dischi magnetici - tecnologia




## Floppy disk

**Dischi flessibili (FD, dischetti):**

- **capacità < 5 MB**
- **rimovibili, racchiusi in un involucro protettivo**

## Hard disk

**Dischi rigidi (HD, "Winchester"):**

- **capacità: 1 GB ... 1 TB**
- **velocità di trasferimento: O(10 MB/s)**
- **fissi, racchiusi in un contenitore in cui è fatto il vuoto**
- **velocità di rotazione:**
  - **10,000 RPM (dischi veloci, per server)**
  - **5,000 RPM (per PC e WS)**
  - **3,000 RPM (per computer portatili)**

© A.Lioy - Politecnico di Torino (1992-2008)

## Dischi RAID

**Invece di costruire dischi che non si guastano (che costerebbero cari) si usano dischi di qualità inferiore con meccanismi di correzione degli errori misti HW/SW:**

**R Redundant**
**A Array of**
**I Independent (Inexpensive)**
**D Disks**

## Livelli RAID

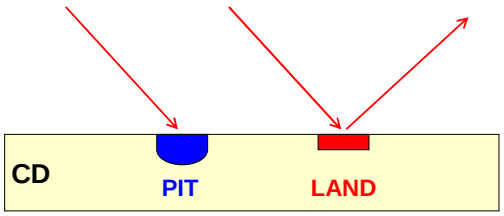
| | |
|---|---|
| **RAID-0** | **disk striping**<br>**velocità ma nessuna protezione** |
| **RAID-1** | **disk mirroring**<br>**protezione al costo del 50%** |
| **RAID-3** | **disk byte-striping**<br>**un disco dedicato a ECC** |
| **RAID-4** | **disk block-striping**<br>**un disco dedicato a ECC** |
| **RAID-5** | **disk block-striping**<br>**ECC distribuito su tutti i dischi** |

## CD (Compact Disc)

- **dischi ottici, a sola/prevalente lettura**
- **singola faccia, rimovibili**
- **capacità: 600 MB**
- **lenti: tempo di accesso 300 ms**
- **uso: distribuzione di software o dati**
- **tipologie:**
  - **CD-ROM (dati registrati in produzione)**
  - **CD-R (dati registrabili da utente)**
  - **CD-RW (dati registrabili e cancellabili da utente, ma con perdita di capacità)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Tecnologia dei CD ottici




## Velocità dei drive per CD-ROM

**Velocità di trasferimento:**

- **base = 150 KB/s**
- **oggi 2x, 3x, 4x, 8x, 16x, ..., 52x**

**Quando sono indicate tre velocità esse sono nell'ordine:**

- **write-once / rewrite / read**
- **es. 12x / 10x / 32x (= 1.8 / 1.5 / 4.8 MB/s)**

**Tempo di accesso:**

- **O ( 100 ms )**

## DVD (Digital Versatile Disc)

- **anche Digital Video Disc perché molto usati per i filmati (in formato MPEG-2)**
- **dischi con diametro 8 o 12 cm**
- **single side (SS) o double side (DS)**
- **single layer (SL) o double layer (DL)**
- **2 GB = 1 ora di video ad alta qualità, con 3 colonne sonore differenti (durata ancora maggiore se video con qualità VHS)**
- **velocità 1x = 1.32 MB/s**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Formati DVD (8 cm)

- DVD-1 (8cm, SS/SL) = 1.36 GB
- DVD-2 (8cm, SS/DL) = 2.48 GB
- DVD-3 (8cm, DS/SL) = 2.72 GB
- DVD-4 (8cm, DS/DL) = 4.95 GB
- DVD-R (8cm, SS/SL) = 1.15 GB
- DVD-R (8cm, DS/SL) = 2.29 GB

## Formati DVD (12 cm)

- DVD-5 (12cm, SS/SL) = 4.7 GB
- DVD-9 (12cm, SS/DL) = 8.5 GB
- DVD-10 (12cm, DS/SL) = 9.4 GB
- DVD-18 (12cm, DS/DL) = 17.0 GB
- DVD-R (12cm, SS/SL) = 4.7 GB
- DVD-R (12cm, DS/SL) = 8.5 GB
- DVD-RAM (12cm, SS/SL) = 2.40 GB
- DVD-RAM (12cm, DS/SL) = 4.80 GB

## Tipi di DVD

**DVD-video**
**DVD-ROM**
argento
read-only

**DVD-R**
rosa
recordable
(write-once)

**DVD-RAM**
involucro
doppia
faccia
(read-write)

© A.Lioy - Politecnico di Torino (1992-2008)

## Formati in competizione

**DVD Forum (www.dvdforum.org)**

- **formati DVD-**

**DVD+RW Alliance (www.dvdrw.com)**

- **Dell, HP, Verbatim, Philips, Ricoh, Sony, Thomson, Yamaha**
- **formati: DVD+**
- **maggiore compatibilità coi lettori esistenti**

## Blue-ray DVD (BD)

- **nuovo formato DVD (incompatibile col precedente)**
- **usa laser blu-violetto ($\lambda$=405 nm) invece che rosso ($\lambda$=650 nm)**
- **25 / 50 GB (single / double layer)**
- **richiesto per memorizzare un film in HD**
- **http://www.bluraydisc.com**

## I nastri

- **striscia plastica avvolta su bobina**
- **accesso sequenziale**
- **tempo di accesso: O(60 s)**
- **tecnologia magnetica o ottica**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Nastri magnetici

- densità di registrazione
  1600-6250 BPI ( Bit-Per-Inch )
- 9 tracce parallele
- capacità: secondo la lunghezza
- uso: backup
- velocità: variabile (fino a 1 GB/h)

## DDS (Digital Data Storage)

- evoluzione del DAT
- cassette da 4 mm con tracce elicoidali
- DDS-1 (1989) = 2 GB, 0.4 MB/s, 90 m
- DDS-2 (1993) = 4 GB, 0.5 MB/s, 120 m
- DDS-3 (1996) = 12 GB, 1 MB/s, 125 m
- DDS-4 (1999) = 20 GB, 3 MB/s, 125 m
- DAT 72 (2003) = 36 GB, 3.2 MB/s, 170 m
- DAT 160 (2007) = 80 GB, 6.9 MB/s, 150 m
- capacità raddoppia con compressione

## DLT (Digital Linear Tape)

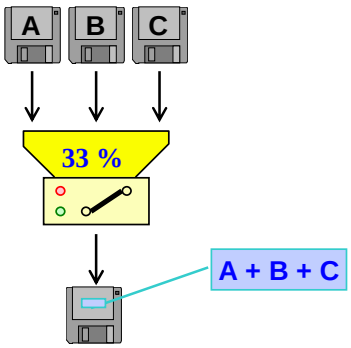
- evoluzione del DAT (Digital Audio Tape)
- cassette da 1/2 inch
- tracce lineari parallele (128 ... 208)
- capacità raddoppia con compressione
- DLT-7000 (1996) = 35 GB, 5 MB/s
- DLT-8000 (1999) = 40 GB, 6 MB/s
- super DLT usa traccia ottica per sincro
- SDLT-600 (2004) = 300 GB, 36 MB/s
- DLT-S4 (2006) = 800 GB, 60 MB/s

*© A.Lioy - Politecnico di Torino (1992-2008)*

## LTO (Linear Tape Open)

- formato “open” (contro DLT e AIT)
- cassette modello “Ultrium” (1/2 inch)
- capacità hw di compressione, cifratura e WORM
- LTO-1 (2000) = 100 GB, 20 MB/s
- LTO-2 (2003) = 200 GB, 20 MB/s
- LTO-3 (2005) = 400 GB, 20 MB/s
- LTO-4 (2007) = 800 GB, 20 MB/s

## La compressione dei dati



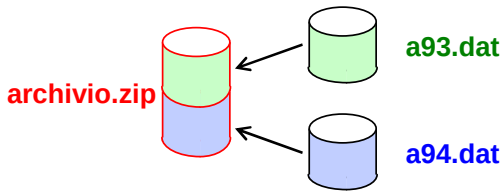

## Compressione dei dati

- si cercano tra i dati le sequenze di bit più frequenti ...
- ... le si sostituisce con sequenze più corte ...
- ... e si memorizza assieme ai dati anche la tabella di conversione
- riduzione tra il 50% (binari) ed il 90% (testo alfanumerico)

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Archiviazione dei dati

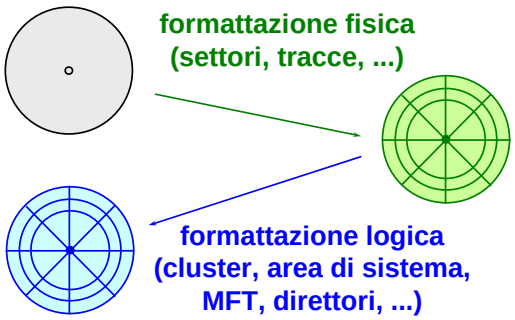
**Talvolta è pratico raccogliere un insieme di file all'interno di un altro file ( archivio )**




## Programmi di compressione e/o archiviazione

| | tipo | DOS/Win | UNIX |
|---|---|---|---|
| zip | C+A | X | X |
| rar | C+A | X | X |
| arj | C+A | X | – |
| compress | C | – | X |
| tar | A | – | X |
| gzip | C | X | X |

## Inizializzazione dei dischi




© A.Lioy - Politecnico di Torino (1992-2008)

## Il file system

- organizzazione logica dei dati sulla memoria di massa
- tipi di file (formato fisso, variabile, binario/stream, con indice, ...)
- organizzazione a direttori, folder, cartelle
- nomi e protezioni dei file
- dipende dal sistema operativo!
- FAT-16, FAT-32, NTFS, UFS, EXT3, ...

## Allocazione file su disco

- file registrati in cluster
- 1 cluster = $2^N$ settori
- dimensione su disco >= quantità dati
- se dim. file >= dim. cluster, allora si occupano altri cluster

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| settori | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 | S9 | S10 |
| cluster | C1 | | C2 | | C3 | | C4 | | C5 | |
| file | a.txt (1) | | b.txt | | *free* | | a.txt (2) | | *free* | |
| dati | 0101 | | 0011 | | | | 11.. | | | |

## Settori e cluster (NTFS)

50.334.416 settori da 512 B ciascuno
( dim disco ~ 25 GB )

```
> fsutil fsinfo ntfsinfo C:
NTFS Volume Serial Number : 0x76bdf8890fb5e386
Version :                   3.1
Number Sectors :            0x0000000003000ad0
Total Clusters :            0x000000000060015a
Free Clusters  :            0x00000000000c073f
Total Reserved :            0x0000000000000000
Bytes Per Sector  :         512
Bytes Per Cluster :         4096
. . .
```

8 settori per cluster = 4 kB
( occupazione minima su disco )

© A.Lioy - Politecnico di Torino (1992-2008)

## Frammentazione

- **frammentazione se cluster non contigui**
- **alta frammentazione comporta alti tempi di lettura/scrittura**
- **programmi di deframmentazione**
- **percentuale di frammentazione:**
  **F = (n.frammenti – 1) / n.cluster**
- **es. file con 3 cluster in 2 frammenti**
  **F = (2 – 1) / 3 = 33%**
  **ossia in media un accesso su tre richiede seek**

## Esercizio

**Un HD contiene un file da 1 GB, memorizzato su cluster contigui, senza frammentazione. Sapendo che il disco ha tempo di accesso di 20 ms e velocità di trasferimento di 32 MB/s, si determini il tempo necessario a leggere l'intero file.**

**Soluzione:**

- **T = 20 ms + 1 GB / 32 MB/s**
  **= 20 ms + 32 s**
  **= 32.02 s**

## Esercizio

**Un HD contiene un file da 1 GB, completamente frammentato. Sapendo che il disco ha cluster da 512 byte, tempo di accesso di 20 ms e velocità di trasferimento di 32 MB/s, si determini il tempo necessario a leggere l'intero file.**

**Soluzione:**

- **T = 20 ms x $2^{20}$ KB / 0.5 KB**
  **+ 1 GB / 32 MB/s**
  **= 10485.76 s + 32 s**
  **= 10517.76 s ~ 2.9 ore**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Memorie SD

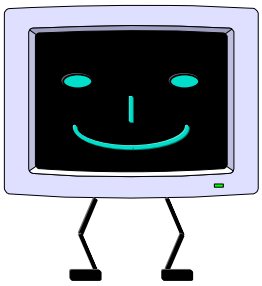
- **Secure Digital (www.sdcard.org)**
- **alta capacità (4 GB, in crescita)**
- **no parti mobili (basso consumo energia)**
- **dimensioni ridotte (24x32 mm, o miniSD 20x21.5 mm)**
- **veloce (100 Mbps 4-bit mode, 25 Mbps 1-bit SPI mode)**
- **dispositivo attivo (protezione del copyright - Secure Digital Music Initiative, SDMI)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

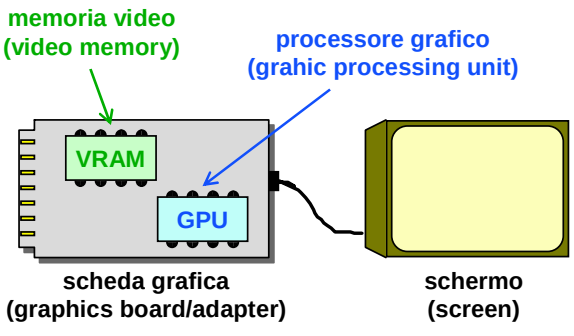
## Dispositivi di I/O

- **I/O = Input / Output**
- **I/U = Ingresso / Uscita**
- **dispositivi standard = basso costo (video, stampante, tastiera, ...)**
- **dispositivi non standard = alto costo (sensori di temperatura, ...)**

## Dispositivi di visualizzazione



## Schema funzionale di una VDU ( Video Display Unit )




*© A.Lioy - Politecnico di Torino (1992-2008)*

## Parametri di confronto

- **capacità grafiche**
- **risoluzione**
- **colori**
- **dimensione**

## Capacità grafiche

**VDU alfanumerica:**

- **solo caratteri**
- **grafica rudimentale tramite *caratteri grafici***

**VDU grafica:**

- ***pixel*** **(minimo elemento grafico controllabile individualmente); anche detto *punto* ( *dot* )**

## Risoluzione della VDU

**VDU alfanumerica:**

- **righe × colonne**
- **tipicamente 25 × 80**

**VDU grafica:**

- **$pixel_H \times pixel_V$**
- **da 640×480 a 3200×2400**
- ***dot pitch*** **= distanza tra due pixel (minore è il pitch, migliore la visione)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## VDU a colori

**VDU monocromatica:**

- **fondo nero, pixel colorato**
- ***reverse* video**

**VDU a colori:**

- **colori RGB (Red, Green, Blue)**
- **N colori simultaneamente**
- **P colori nella tavolozza ( *palette* )**
- **P ≥ N**

## La memoria video

**Serve a memorizzare l'immagine (perché deve essere continuamente riscritta sullo schermo = *refresh*):**

- **caratteri e loro attributi**
- **pixel e loro colore**

## La memoria video

***Domanda*: quanta memoria video occorre per visualizzare un'immagine 640×480 a 256 colori?**

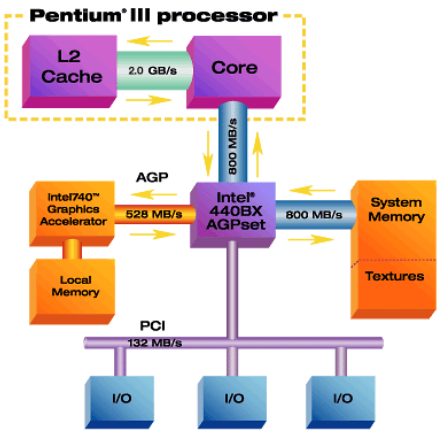
***Soluzione*:**
**640 × 480 = 307,200 pixel**
**307,200 : 1024 = 300 Kpixel**
**$\lceil \log_2 256 \rceil$ = 8 bit/pixel = 1 byte/pixel**

***Risposta*:**
**300 Kbyte**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## AGP (Accelerated Graphics Port)

- **collegamento diretto tra chip grafico e memoria di sistema:**
  - **non intasa il bus dei periferici**
  - **alta velocità (~100 MB/s)**
- **evoluzione di PCI 66 MHz per la grafica:**
  - **AGP = 66 MHz (264 MB/s)**
  - **AGP 2x = 133 MHz (528 MB/s)**
  - **AGP 4x = 266 MHz (1056 MB/s)**
- **oggi quasi completamente sostituito da PCI-X**




## Dimensioni dello schermo

**E' la misura della diagonale:**

- **10 ÷15" (computer portatili)**
- **15÷17" (computer da scrivania)**
- **17÷21" (computer professionali)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Tecnologia degli schermi

- **a fosfori (CRT, Cathode Ray Tube)**
- **a cristalli liquidi (LCD, Liquid Crystal Display)**
- **a matrice attiva (TFT, Thin Film Transistor)**

## Schermi a fosfori

- **tecnologia uguale agli schermi TV**
- \+ **grandi dimensioni**
- \+ **ottima risoluzione**
- \+ **ottima resa colorimetrica**
- – **elevato consumo elettrico**
- – **pesanti ed ingombranti**
- – **emettono radiazioni**

## Schermi a cristalli liquidi

- **a matrice passiva**
- **DSTN (Double Scan Twisted Nematic)**
- \+ **leggeri e sottili**
- \+ **basso consumo elettrico**
- – **dimensioni limitate**
- – **bassa risoluzione**
- – **resa colorimetrica mediocre**
- – **alta persistenza**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Schermi TFT

- **a matrice attiva**
- **Thin Film Transistor**

+ **leggeri e sottili**
+ **ottima risoluzione**
+ **buona resa colorimetrica**

– **consumo elettrico medio**
– **dimensioni limitate**
– **costo elevato**

## VDU per PC (primi modelli)

| *adattatore* | *risoluzione* | *colori* |
|---|---|---|
| CGA | 320 x 200 | 4 |
| | 640 x 200 | 2 |
| EGA | 320 x 200 | 16 |
| | 640 x 200 | 16 |
| | 640 x 350 | 64 |
| VGA | 320 x 200 | 256 |
| | 640 x 480 | 16 |

## VDU per PC

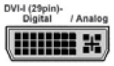
| *adattatore* | *risoluzione* | *pixel* |
|---|---|---|
| SVGA | 800 x 600 | |
| XGA | 1024 x 768 | 0.8 M |
| SXGA | 1280 x 1024 | 1.3 M |
| UXGA | 1600 x 1200 | 1.9 M |
| QXGA | 2048 x 1536 | 3.1 M |
| QSXGA | 2560 x 2048 | 7.6 M |
| QUXGA | 3200 x 2400 | 7.7 M |
| QUXGA-W | 3840 x 2400 | 9.2 M |

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Connettori scheda grafica - monitor

- connettore D-SUB (anche detto VGA)
- 15 pin, solo segnale analogico

- connettore DVI-D (Digital Video Interface)
- 24 pin, solo segnale digitale




- connettore DVI-I
- 29 pin = 24 pin segnale digitale (DVI-D) + 5 pin segnale analogico
- analogico / digitale mutuamente esclusivo

## Acceleratori grafici

- **schede grafiche con un processore (GPU) che realizza in HW operazioni grafiche complesse**
- **acceleratori 2D/3D**
- **occorre che il SW sia a conoscenza di questo HW speciale!**

## Dispositivi di stampa

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Categorie di dispositivi

- **stampanti (a caratteri, grafiche)**
- **plotter**
- **fotoriproduttori (foto, dia)**
- **macchine per fotocomposizione**

## Parametri di valutazione

- **velocità di stampa**
- **capacità grafiche**
- **risoluzione**
- **formato della carta**
- **caratteri di stampa (font, punti, formati)**
- **colori**

## Velocità di stampa

**Unità di misura:**

- **per stampanti "a caratteri"**
  - **CPS ( Characters Per Second )**
- **per stampanti "di linea"**
  - **LPS ( Lines Per Second )**
  - **LPM ( Lines Per Minute )**
- **per stampanti grafiche (o "di pagina")**
  - **PPM ( Pages Per Minute )**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Risoluzione di stampa

- **solitamente specificata solo per stampanti con capacità grafiche.**
- **si misura in DPI (Dots Per Inch)**
- **si ricordi 1 inch = 2.54 cm**
- **per stampanti solo alfanumeriche si può misurare in CPI (Characters Per Inch)**

## Formato della carta

- **foglio singolo, modulo continuo, buste, rullo**
- **tutto il mondo (tranne USA) usa il formato ISO 216:**
  - **altezza = base × sqrt(2)**
  - **A0, A1, A2, A3, A4 (210 × 297 mm), …**
  - **ogni foglio è il doppio del precedente (es. A3 = 2 A4)**
- **USA: letter (8.5 × 11” = 216 × 279 mm), legal (8.5 × 14” = 216 × 356 mm), ledger/tabloid, ...**

## Spaziatura dei caratteri

- **spaziatura fissa**

```
buon lavoro a
tutti quanti!
```

- **spaziatura proporzionale**

**buon lavoro a**
**tutti quanti!**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Le famiglie di caratteri ( *font* )

**Helvetica**

**Times ( New Roman )**

*Lucida Handwriting*

`Courier`

## Lo stile dei caratteri

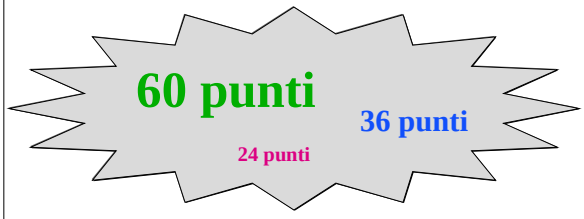
Testo normale

**Testo in grassetto ( bold )**

*Testo in italico*

***Testo in italico-grassetto***

## Dimensione dei caratteri




**Font fissi: solo alcune dimensioni**
**Font scalabili: qualunque dimensione**
**Punto tipografico = 1/72 di inch**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Colori

**Stampa bianco e nero:**

- **bianco e nero puro**
- **toni di grigio**

**Stampa a colori:**

- **tricromia (CMY = Cyan, Magenta, Yellow)**
- **quadricromia (CMYK = CMY + nero!)**

## Colori: RGB o CMY(K)?

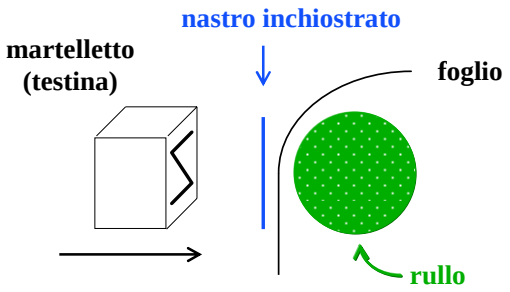
- **notazioni teoricamente equivalenti**
- **RGB = metodo additivo, per video**
  - **nero = assenza di luce (schermo spento)**
  - **colori (Red, Green, Blue) = emissione di luce**
- **CMY(K) = metodo sottrativo, per stampa**
  - **bianco = sfondo di stampa (foglio vuoto) che riflette tutta la luce**
  - **colori (Cyan, Magenta, Yellow) = inchiostro che riflette solo alcune frequenze**
  - **K (=black) per più qualità e meno consumo**

## Tecnologia delle stampanti

- **stampanti a impatto**
- **stampanti termiche**
- **stampanti a getto d'inchiostro**
- **stampanti laser**
- **stampanti a sublimazione**

*© A.Lioy - Politecnico di Torino (1992-2008)*

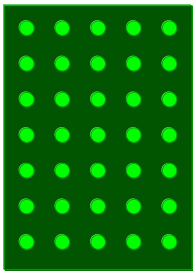
## Stampanti a impatto




## Stampanti ad aghi

- 80 ... 160 CPS
- capacità grafiche (limitate)
- costo ~ 50 Euro

testina = matrice di aghi



## Stampanti a catena ( *line printer* )

- stampano una riga per volta
- 1200 LPM (draft), 900 LPM (standard), 500 LPM (lettera)
- costo ~ 1000 Euro

| | | | | | |
|---|---|---|---|---|---|
| C | I | A | O | | ! |
| D | J | B | P | , | ? |
| E | K | C | Q | ; | / |

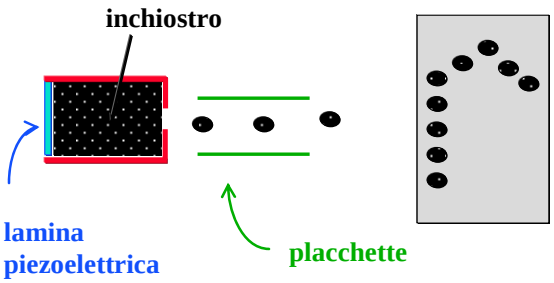
catene

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Stampanti termiche

- **medesimo principio delle stampanti ad aghi**
- **aghi riscaldati elettricamente per impressionare una carta termosensibile**
- **non richiedono inchiostro**
- **per piccole stampanti (es. registratori di cassa)**

## Stampanti a getto d'inchiostro



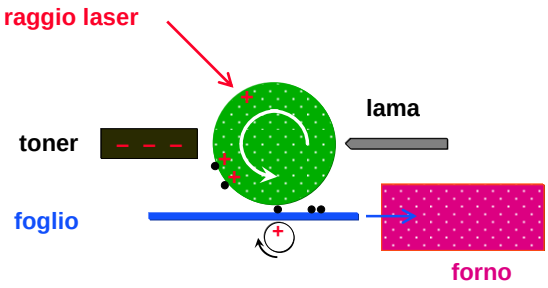

## Stampanti a getto d'inchiostro

- **B/N e colori**
- **80 ... 160 CPS**
- **capacità grafiche ( 300…600 DPI )**
- **costo ~ 100 Euro**

\+ **ottimo rapporto qualità - prezzo**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Stampanti laser




## Stampanti laser

- **B/N**
- **a colori (un passo / quattro passi):**
  - **più costose (soprattutto se a un passo)**
  - **resa cromatica inferiore a quelle a getto d'inchiostro**
- **10 PPM**
- **capacità grafiche ( 600…2400 DPI )**
- **costo ~ 500 Euro**

\+ **ottima qualità e buona velocità**

## Stampanti a sublimazione

- **deposito di cere fuse**
- **a colori**

\+ **altissima resa cromatica**

– **costo esorbitante (5000 Euro)**

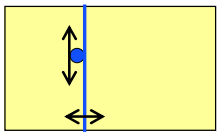
*© A.Lioy - Politecnico di Torino (1992-2008)*

## Plotter

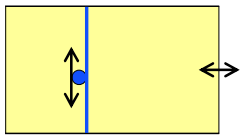
- **emulazione del disegno manuale tramite una testina mobile dotata di penne o a getto di inchiostro**
- **molto usato per disegni di grande formato (es. ingegneria civile)**
- **per disegni di piccolo formato (A4/A3) soppiantato dalle stampanti laser o a getto**

## Plotter - funzionamento

- **foglio fisso, testina mobile X-Y**



- **foglio mobile (X), testina mobile (Y)**



## Il linguaggio Postscript

- **trademark Adobe Inc.**
- **descrive il contenuto di una pagina di output**

\+ **standard de-facto per le stampanti di qualità**

\+ **molto potente: testo e grafica**

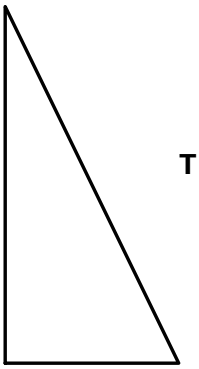
– **lento: è un vero e proprio linguaggio**

– **complesso: richiede CPU e memoria RAM**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Un esempio in linguaggio Postscript

```
100 100 moveto
100 300 lineto
200 100 lineto
closepath
stroke
/Courier findfont
18 scalefont
setfont
200 200 moveto
(Triangolo!) show
```



Triangolo!

## L'interprete Postscript

- **una stampante Postscript deve contenere una CPU, memoria ed istruzioni per eseguire il programma Postscript ricevuto**
- **esistono traduttori da Postscript ai formati di altre stampanti ( es. ghostscript )**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il linguaggio PCL

- **Printer Control Language**
- **trademark Hewlett-Packard**
- **vari livelli (PCL1 ... PCL6)**
- **più semplice del Postscript ma meno generale**
- **standard per le stampanti più semplici ed economiche**

## PDF

- **trademark Adobe**
- **Portable Document Format**
- **evoluzione di Postscript (ad esempio, inserimento di link verso il web)**
- **visualizzabile e stampabile col prodotto (gratuito) Acrobat Reader**

## Dispositivi di input

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Categorie di dispositivi

**Input a caratteri:**

- **tastiera**

**Puntatore ( *point-and-click* ):**

- **mouse, video sensibile, tavoletta grafica**

**Scansione di immagine:**

- **scanner**
- **lettori ottici e magnetici**

## La tastiera ( *keyboard* )

- **caratteri alfanumerici**
- **tasti funzione e caratteri di controllo**
- **numero e disposizione dei tasti ( *layout* ) variabili**
- **modelli *ruggerized* per ambienti industriali**

## Il mouse

- **dispositivo di puntamento (rileva movimento *relativo* e per analogia fa muovere il *cursore* video)**
- **due o tre tasti (per puntamento e selezione)**
- **modelli meccanici (più robusti, meno precisi)**
- **modelli ottici (più precisi, meno robusti)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il video sensibile ( *touchscreen* )

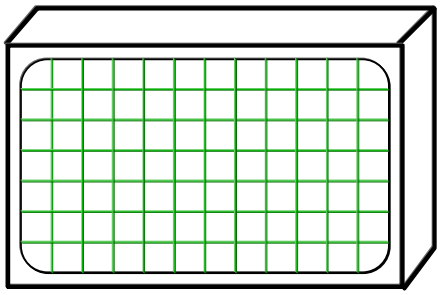
- **aree sensibili (coordinate cartesiane)**
- **effetto capacitivo (più sensibile)**
- **effetto fotoelettrico (errori di parallasse)**

\+ **ottimo per uso non specialistico in aree pubbliche**

– **bassa risoluzione**

– **errori di parallasse / sporcizia**

## Video sensibile - funzionamento



## La tavoletta grafica

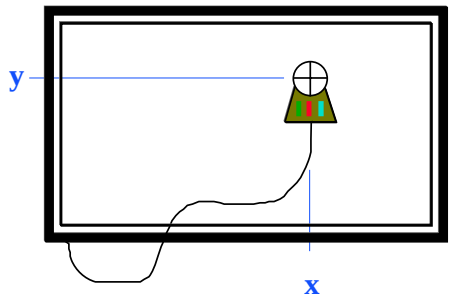
- **misura di coordinate assolute**
- **riferimento cartesiano**
- **rilevamento di punti singoli o tratti continui**

\+ **retroilluminabile (radiografie)**

– **errori umani di posizionamento**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Tavoletta grafica - funzionamento




## Lo scanner

- **dispositivo di acquisizione immagini (*pixmap*)**
- **risoluzione (DPI)**
- **formato (A3, A4)**
- **bianco/nero, toni di grigio, colori**

\+ **grande precisione**

– **necessita di appositi programmi per riconoscimento di testo (OCR, Optical Character Recognition) disegni**

## Scanner - funzionamento

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Scanner - memoria richiesta

***Quanta memoria occupa un'immagine 5"×3" acquisita tramite uno scanner a 300 DPI, 256 toni di grigio?***

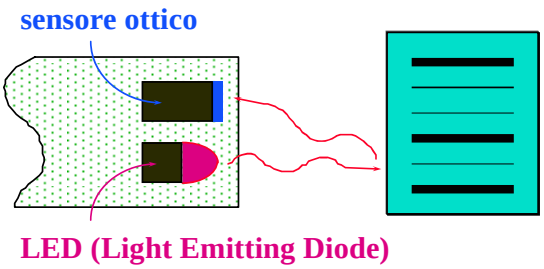
**300 DPI × 5" = 1500 D**
**300 DPI × 3" = 900 D**
**1500 × 900 = 1,350,000 pixel**
**256 toni → 8 bit/pixel = 1 byte/pixel**
**1,350,000 byte ≅ 1.3 MB**

## La penna ottica

- **variazioni della luce riflessa**
- **lettura di codici a barre (EAN)**

\+ **leggera, facile uso**
– **superfici pulite, stampe nitide**

## Penna ottica - funzionamento




*© A.Lioy - Politecnico di Torino (1992-2008)*

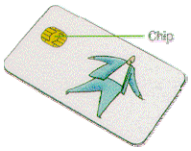
## Carte elettroniche

**Carte passive:**

- **magnetiche**
- **a memoria elettronica**

**Carte attive ( *smart card* ):**

- **carte a chip (= microprocessore + memoria)**

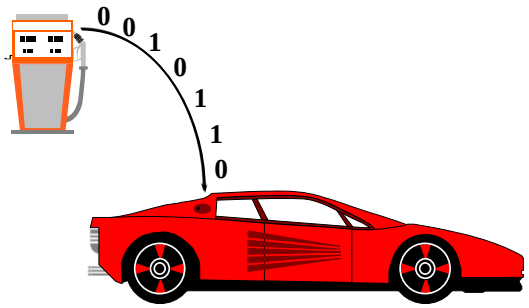



## Identificazione radio

- **radio tag, RFID**
- **esempi:**
  - **gestione spesa supermercato**
  - **inventario automatico magazzino**
- **servizi *location-based* (tel. cellulare):**
  - **tracciamento persone**
  - **offerte speciali per chi si trova in una certa area**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il software




## Definizioni

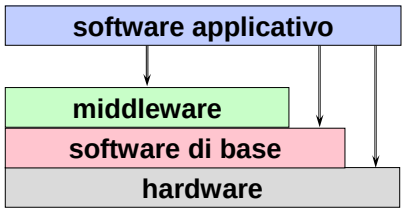
**Software**

***l'insieme di dati e programmi che permettono ad un elaboratore di svolgere le sue funzioni***

**Firmware**

***un software essenziale per il funzionamento di uno strumento ed indissolubilmente legato ad esso; solitamente memorizzato in ROM***

## Classificazione del software




*© A.Lioy - Politecnico di Torino (1992-2008)*

## Linguaggi di programmazione

**Servono a tradurre gli algoritmi in istruzioni per l'elaboratore.**

**Ne esistono varie categorie:**

- **linguaggi macchina**
- **linguaggi assembler**
- **linguaggi ad alto livello**

## Linguaggio macchina

- **codifica binaria delle istruzioni**
- **molto efficiente**
- **dipende dalla CPU (registri, operazioni, ...)**
- **dipende dall'architettura del calcolatore (quantità di memoria, modalità di I/O, ...)**

## Linguaggio assembler

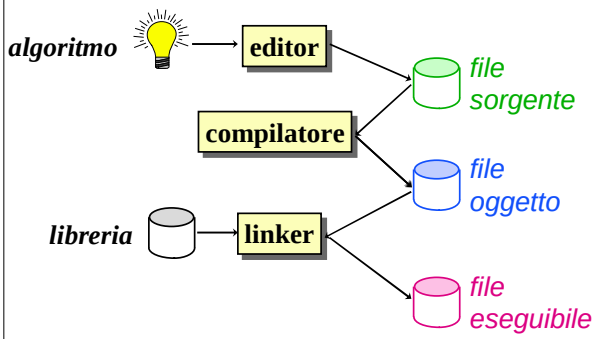
- **codifica simbolica delle istruzioni**
  ADD R1, R2, M[R4] ; R1= R2+ M [ R4 ]
- **meta-istruzioni**
  TOT: DB 0 ; crea ed azzera TOT
- **linguaggio unico per architettura di CPU (es. Dec AXP, Intel 80x86)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

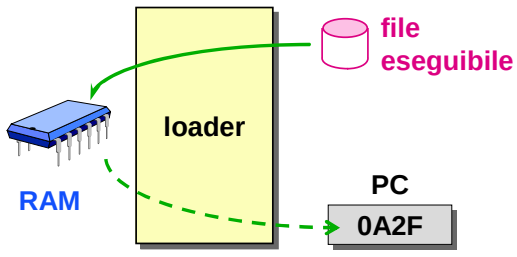
## Linguaggi ad alto livello (HLL - High Level Language)

- **elaboratore virtuale**
- **istruzioni orientate alla soluzione di problemi**
  A = SINH (5.57*C)
- **linguaggio indipendente dalla piattaforma HW (→ traduttori!)**

## Traduzione dei linguaggi




## Esecuzione di un programma

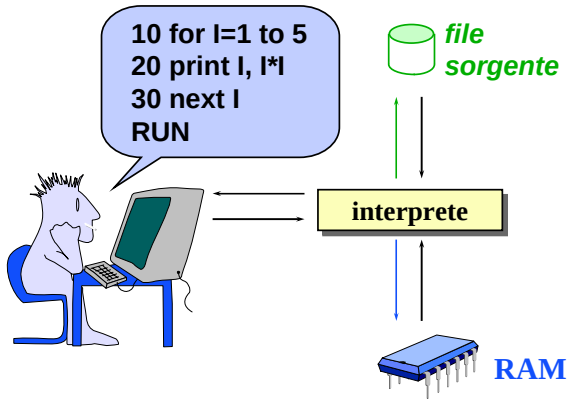



© A.Lioy - Politecnico di Torino (1992-2008)

## Debugger

**Sostituisce il loader per l'esecuzione controllata di un programma (per cercarne i *bug*):**

- **single-step**
- **breakpoint**
- **tracepoint**
- **visualizza/cambia il valore delle variabili**




## Alcuni interpreti

- **interprete di un linguaggio di programmazione (Basic, Java, …)**
- **interprete dei comandi di un sistema operativo (**command.com **in MS-DOS,** cmd.exe **in Win32, le *shell* UNIX, ...)**
- **i fogli elettronici (Lotus 1-2-3, Excel, QuattroPro, ...)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Interprete o compilatore?

**Interprete:**

- **controlla e traduce le istruzioni ad ogni esecuzione del programma**

\+ **correzione degli errori e debug veloce**

– **caricamento ed esecuzione lenta**

– **codice sorgente leggibile**

## Interprete o compilatore?

**Compilatore (+ linker + loader):**

- **controlla e traduce le istruzioni una sola volta**

\+ **caricamento ed esecuzione veloce**

\+ **codice sorgente non leggibile**

– **correzione degli errori e debug lenti**

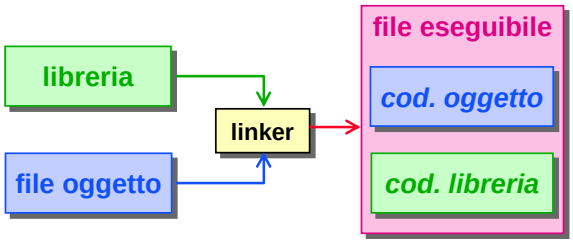
## Sistemi integrati di programmazione

- **IDE (Integrated Development Environment)**
- **uniscono editor, compilatore, linker e debugger**
- **hanno rimpiazzato gli interpreti**
- **Borland:**
  **TurboC, C++Builder, Delphi, ...**
- **Microsoft:**
  **VisualC, VisualBasic, ...**

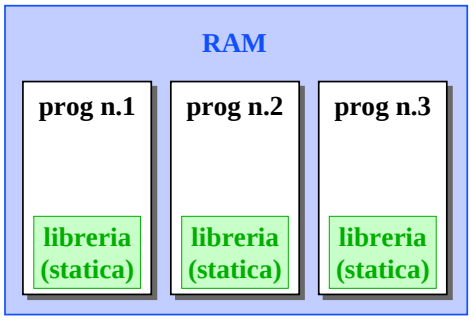
*© A.Lioy - Politecnico di Torino (1992-2008)*

## Librerie statiche

**Il codice delle funzioni di libreria necessarie viene inserito nel file eseguibile:**




## Librerie statiche

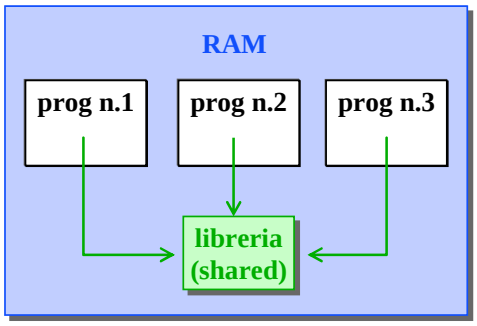



## Librerie dinamiche (o condivise)

- **anche note come DLL (Dynamic Link Library) oppure *shared library***
- **il codice delle funzioni di libreria necessarie NON viene inserito nel file eseguibile ma viene solo messo un riferimento**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Librerie dinamiche




## Librerie statiche o dinamiche?

**Vantaggio delle librerie dinamiche:**

- **ridotta dimensione dell'eseguibile (=spazio su disco, velocità di caricamento, occupazione di RAM)**

**Vantaggio delle librerie statiche:**

- **autoconsistenza (=portabilità)**

## Portabilità del software

**Per *portabilità* si intende la capacità di un modulo software di essere eseguito su piattaforme (hw/sw) diverse da quella su cui è stato sviluppato.**

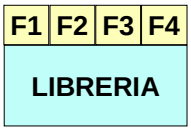
**In generale la portabilità del software è molto scarsa ed è stata *parzialmente* raggiunta solo in alcuni ambiti (es. PC Intel 80x86 con MS-Windows).**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## API (Application Programming Interface)

**Interfaccia standard di una libreria:**

- **chiamate di funzioni**
- **formato dei parametri**




**Se c'è compatibilità di API, allora per portare un programma basta trasportare il codice sorgente e ricompilarlo nel nuovo ambiente operativo.**

## ABI (Application Binary Interface)

**Specifica per garantire la portabilità a livello binario (file eseguibile):**

- **formato del file eseguibile**
- **organizzazione dell'I/O**
- **chiamata alle funzioni di sistema**
- **(stesso linguaggio macchina)**

**Se c'è compatibilità a livello ABI, allora un file eseguibile può essere trasportato e direttamente eseguito nel nuovo ambiente operativo.**

## Il linguaggio FORTRAN

- **FORmula TRANslation (1956)**
- **calcoli tecnico-scientifici**
- **grande disponibilità di librerie (es. NAG, IMSL)**
- **Fortran-II, Fortran-IV**
- **Fortran-77 (prog. strutturata)**
- **Fortran-90 (parallelo/vettoriale)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il linguaggio COBOL

- **COmmerce and Business Oriented Language (1960)**
- **elaborazione di archivi, gestione di maschere video e tabulati**

## Il linguaggio BASIC

- **Beginner's All-purpose Symbolic Instruction Code (1962)**
- **molto semplice (o semplicistico?)**
- **capacità grafiche**
- **fornito assieme al S.O. (su PC)**
- **evoluto (?) a VisualBasic:**
  - **linguaggio proprietario di Microsoft**
  - **orientato agli oggetti ed alla grafica**
  - **RAD (Rapid Application Development)**

## Il linguaggio PASCAL

- **N.Wirth (1972)**
- **linguaggio strutturato**
- **completo di tutti i concetti base della programmazione**
- **molto usato per la didattica**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il linguaggio C

- **Bell Labs (anni '70)**
- **linguaggio strutturato, con eccezioni**
- **istruzioni per facilitare l'ottimizzazione**
- **molto efficiente**
- **usato nella programmazione di sistemi (compilatori, sistemi operativi, ...)**

## Il linguaggio C++

- **Bell Labs (anni '80)**
- **linguaggio orientato agli oggetti**
- **è un soprainsieme del C**
- **efficiente**

## Il linguaggio Java

- **Sun Microsystems (1995)**
- **linguaggio orientato agli oggetti, simile al C++ ma …**
  - **modello ad oggetti più semplice**
  - **meno costrutti di basso livello (C-like)**
- **bytecode = programma compilato per un computer ideale**
- **JVM (Java Virtual Machine) = interprete che traduce "al volo" il bytecode in codice macchina e lo esegue su un computer reale**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Programmazione procedurale

- **si definiscono i *dati***
- **si definiscono le *procedure* che operano sui dati**
- **se su un determinato dato deve operare più di una procedura, il dato deve essere esterno e globale ad entrambe le procedure**

## Programmazione orientata agli oggetti (OOP)

- ***incapsulamento*: assieme ai dati si specificano tutte le procedure (*metodi*) che possono operare su di essi**
- ***messaggi*: per operare su un dato si invia un messaggio per attivare un metodo**
- ***ereditarietà*: un'oggetto può ereditare tutte le caratteristiche di un altro oggetto e definirne di nuove**

**OGGETTO = DATI + METODI**

## Il linguaggio PROLOG

- **LOGic PROgramming**
- **dati un insieme di *fatti* e di *regole*, ricerca tutti i valori che soddisfano un'*espressione***

*© A.Lioy - Politecnico di Torino (1992-2008)*

## PROLOG: un esempio (regole)

```
figlio(X,Y) :-
   genitore(Y,X).
fratello(X,Y) :-
   genitore(Z,X) & genitore(Z,Y).
nonno(X,Y) :-
   genitore(X,Z) & genitore(Z,Y).
nipote(X,Y) :-
   genitore(X,Z) & genitore(Z,Y).
```

## PROLOG: un esempio (fatti)

```
genitore(andreina,barbara).
genitore(amilcare, barbara).
genitore(barbara,carlo).
genitore(barbara,cecilia).
```

## PROLOG: un esempio (ricerca)

```
nipote(X,amilcare)?
   X = carlo
   X = cecilia
```

*© A.Lioy - Politecnico di Torino (1992-2008)*

## La programmazione

## Sviluppo del software

- **problema**
- **idea (soluzione)**
- **algoritmo (soluzione formale)**
- **programma (traduzione dell'algoritmo in una forma comprensibile da un elaboratore elettronico)**
- **test (su molti casi, con particolare attenzione ai casi limite)**
- **documentazione (manuale utente + manuale del programmatore)**

## Algoritmo

**Un algoritmo può essere considerato un insieme di regole per effettuare un dato compito (risolvere un problema).**

**Un algoritmo deve:**

- **terminare in un tempo finito**
- **produrre un effetto osservabile**
- **essere deterministico, ossia produrre gli stessi risultati a partire dalle stesse condizioni iniziali**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Esecuzione di un algoritmo

**Vengono eseguite in sequenza le operazioni che lo costituiscono.**

**Esistono algoritmi che prevedono:**

- **una sequenza di esecuzione unica**
- **sequenze di esecuzione multiple**

## Esempio: sequenza di esecuzione unica

**Dato il valore di X, calcolare: Y = 5 X + 3**

**Sequenza di esecuzione:**

**1. ricevo il valore di X**

**2. moltiplico X per 5 (sia Z il risultato)**

**3. sommo 3 a Z (sia Y il risultato)**

**4. visualizzo Y**

## Esempio: sequenze di esecuzione multiple

**Dato il valore di X, calcolare la radice quadrata di X+5.**

**Sequenza di esecuzione:**

**1. ricevo il valore di X**

**2. sommo 5 a X (sia Y il risultato)**

**3a. se Y è positivo o nullo, calcolo la sua radice quadrata e la visualizzo**

**3b. se Y è negativo, indico che è impossibile calcolare la sua radice quadrata**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Formalizzazione di una soluzione

- **diagrammi di flusso**
- **pseudo-linguaggio**
- **. . .**

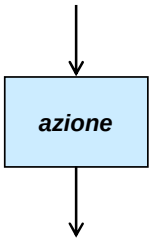
## Diagrammi di flusso (flow-chart)

- **metodo per descrivere in modo formale un algoritmo**
- **blocchi base per descrivere:**
  - **azioni**
  - **decisioni (solo binarie, ossia della logica Booleana)**
- **archi orientati per descrivere la sequenza di svolgimento delle azioni**

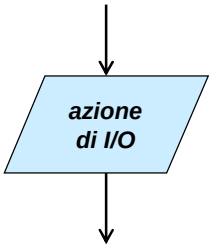
## Blocchi di inizio e fine

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Blocco di azione



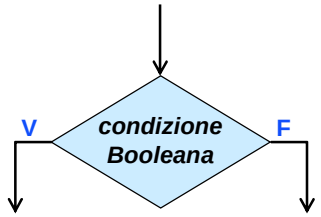

## Blocco di Input/Output



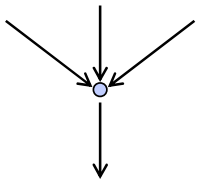

## Blocco di inizializzazione

© A.Lioy - Politecnico di Torino (1992-2008)

## Blocco di decisione binaria




## Connettore



## Regole

- **uno ed un solo blocco START**
- **uno ed un solo blocco STOP**
- **tutti gli archi devono avere origine e fine in un blocco**

© A.Lioy - Politecnico di Torino (1992-2008)

## Diagrammi di flusso strutturati

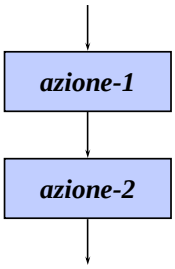
**Un diagramma di flusso è detto *strutturato* se contiene solo un insieme predefinito di strutture:**

- **sequenze**
- **decisioni**
  - **IF-THEN-ELSE**
  - **IF-THEN**
- **cicli**
  - **WHILE**
  - **REPEAT**

## Teorema di Böhm - Jacopini

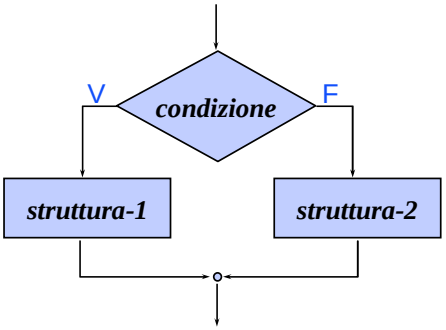
***Qualunque diagramma di flusso è sempre trasformabile in un diagramma di flusso strutturato equivalente a quello dato.***

## Sequenza

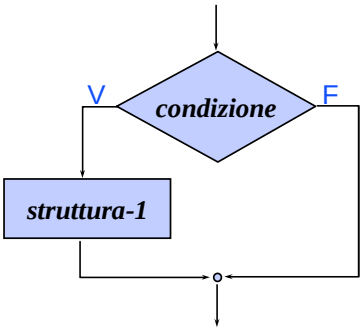



*© A.Lioy - Politecnico di Torino (1992-2008)*

## If-Then-Else




## If-Then

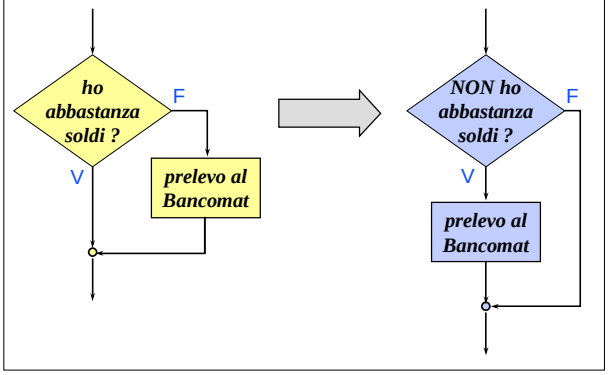



## If-Else

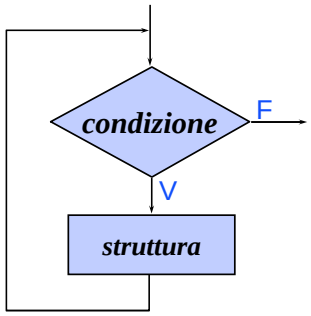
- **non esiste un blocco If-Else …**
- **… perché non è necessario!**
- **basta usare un blocco If-Then in cui la condizione sia stata invertita (negata)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## If-Else (esempio)




## While-Do

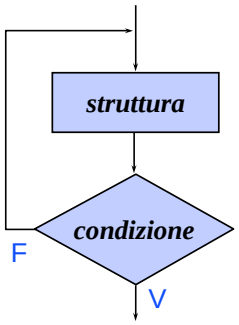



## While-Do

- **la parte ciclica viene eseguita quando la condizione è vera**
- **se abbiamo un ciclo che viene eseguito quando la condizione è falsa, allora occorre trasformarlo in un While-Do mettendo la condizione negata**
- **un ciclo While-Do può essere eseguito zero o più volte**
- **viene eseguito zero volte quando la condizione è subito falsa**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Repeat-Until




## Repeat-Until

- **un ciclo Repeat-Until viene sempre eseguito almeno una volta**

## Verifica di strutturazione

**P1. etichettare ogni blocco**

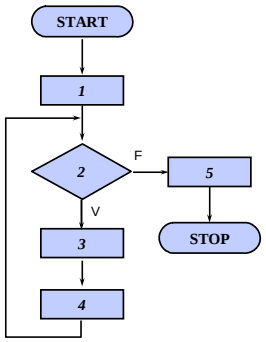
**P2. sostituire ad ogni insieme strutturato un blocco avente come etichetta l'unione delle etichette dei blocchi che lo costituiscono**

**P3. se al passo P2 si è fatta almeno una sostituzione, ripetere il passo P2**

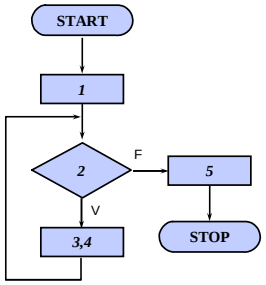
**P4. se alla fine si ottiene un diagramma lineare (una sequenza), allora il diagramma originale è strutturato**

*© A.Lioy - Politecnico di Torino (1992-2008)*

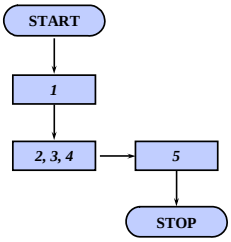
## Esempio: diagramma strutturato




## Esempio: diagramma strutturato

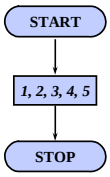



## Esempio: diagramma strutturato

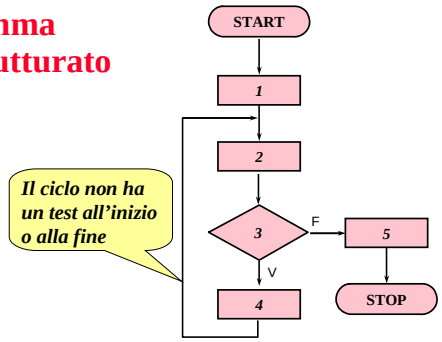



*© A.Lioy - Politecnico di Torino (1992-2008)*

## Esempio: diagramma strutturato

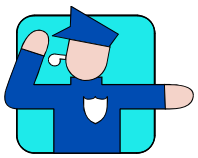



## Esempio: diagramma non strutturato




© A.Lioy - Politecnico di Torino (1992-2008)

## Il sistema operativo



## Il sistema operativo

**Funzionalità svolte:**

- **gestione delle risorse del sistema di elaborazione**
- **interfaccia uomo-macchina (HMI = Human-Machine Interface)**

**Abbreviazioni:**

- **S.O.**
- **O.S. (operating system)**

## S.O. - caratteristiche

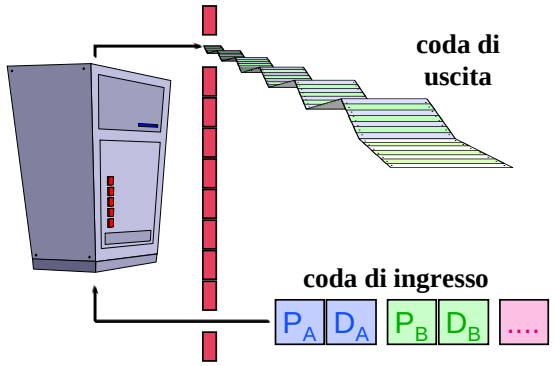
- **gestione della memoria di massa (file system)**
- **gestione della memoria RAM**
- **gestione dei processi**
- **interfaccia utente**
- **n. di utenti simultanei**
- **n. di processi simultanei**

*© A.Lioy - Politecnico di Torino (1992-2008)*

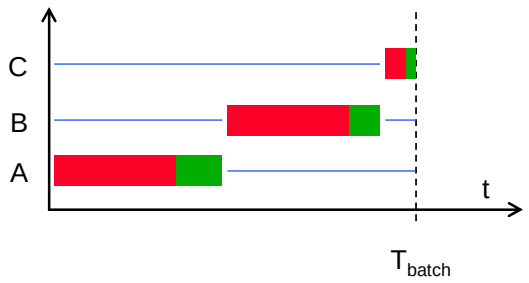
## Sistemi *batch*

- **molto usati negli anni '60**
  **(input = schede perforate)**
  **(output = stampanti a caratteri)**
- **nessuna interazione coll'utente**
  **(input = programma + dati)**
- **avvio dell'elaborazione solo quando un lotto (*batch*) è pieno**
- **elaborazione sequenziale**

## Batch: schema




## Batch: diagramma temporale




*© A.Lioy - Politecnico di Torino (1992-2008)*

## Il batch oggi

**Attualmente non esistono più sistemi completamente batch.**

**Sopravvive una *modalità* batch di elaborazione:**

- **lavoro ( *job* ) = programma + dati**
- **elaborazione sequenziale dei lavori (evita di sovraccaricare l'elaboratore)**

## Un esempio di lavoro *batch*

***Tante persone devono realizzare un collage, avendo a disposizione un solo paio di forbici, un solo tubetto di colla e tanti fogli colorati.***

**Se lavorano tutti simultaneamente si hanno inevitabilmente caos e litigi!**

**Se lavorano uno per volta, si ha ordine ed armonia.**

## Gestione dell'I/O

**Una volta si usava *busy wait (polling)* :**

- **si interroga continuamente il periferico per sapere quando ha terminato**

**Attualmente si usa *spooling* + *interrupt* :**

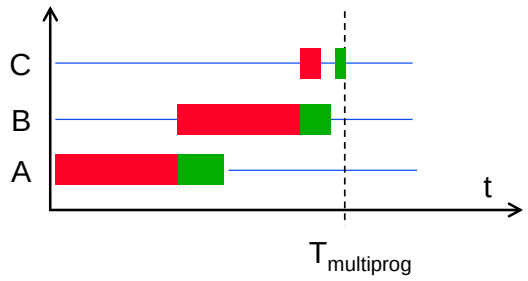
- ***spooling* = i dati da/per i periferici lenti non vengono letti o scritti direttamente ma transitano automaticamente su disco**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Sistemi in multiprogrammazione

- **sfruttano le pause della CPU dovute alla lentezza delle istruzioni di I/O**
- **elaborazione sequenziale con possibilità per i job più piccoli di essere eseguiti nelle pause di quelli più grandi**

## Multiprogrammazione: diagramma temporale




## La multiprogrammazione oggi

**Attualmente non esistono più sistemi in multiprogrammazione pura.**

**Col termine *multitask* si indica la capacità di un S.O. di eseguire più programmi simultaneamente.**

**Se questa capacità manca, si parla di sistema *monotask*.**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Sistemi *time-sharing*

**Anni '70:**

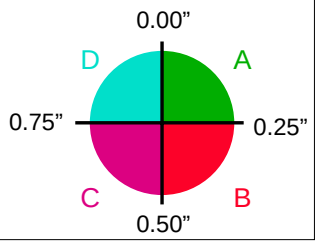
- **nascono le CPU da 1 MIPS**
- **si realizzano i *terminali* (telescriventi, videoterminali)**
- **possibilità di dividere il tempo di CPU tra vari utenti**

## Time-sharing: schema

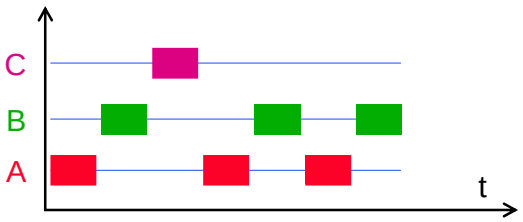
**Ipotesi: 1 MIPS, 4 utenti, 0.25 s/utente**

**Conseguenze:**

- **0.25 MIPS/utente**
- **$T_{ELA} = 4 \times T_{CPU}$**




## Time-sharing: diagramma temporale




© A.Lioy - Politecnico di Torino (1992-2008)

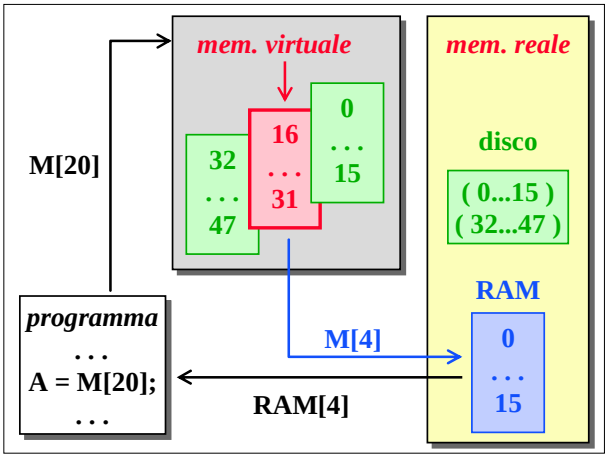
## Il time-sharing oggi

**È il sistema predominante nei grandi sistemi (per motivi economici):**

- **multitask**
- **multiutente**

## La memoria virtuale

- **la dimensione dell'A-BUS limita superiormente la quantità di memoria indirizzabile direttamente ...**
- **... ma non è detto che questa sia tutta fisicamente presente !**
- **esempio: 80486 ha ABUS da 32 bit, ossia 4 GB teorici, ma solitamente solo qualche MB di memoria fisica**




*© A.Lioy - Politecnico di Torino (1992-2008)*

## Lo *swap*

- **i dati dei programmi non in esecuzione (ed anche i dati non immediatamente utili dei programmi in esecuzione) possono essere tolti dalla memoria centrale e parcheggiati su disco nell'*area di swap***
- **swap : RAM = 3 : 1 (max)**
- **max. memoria accessibile = max (RAM, area di swap)**

## Protezione della memoria

**Il S.O. deve garantire che la memoria usata da un processo non possa essere letta o modificata da un altro processo.**

**Nei S.O. meno evoluti (MS-DOS, MS-Windows) questo non capita:**

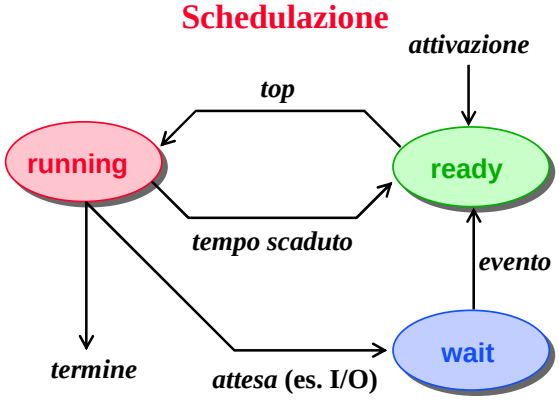
- ***crash* del sistema**
- ***virus***

## Lo scheduling (pianificazione)

- **componente dei S.O. multitasking**
- **politica di sequenzializzazione dei processi**
- **classi + priorità (fissa o variabile)**
- **gestita dallo schedulatore ( *scheduler* )**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Schedulazione




## Sistemi *real-time*

- **in grado di attivare un processo entro breve tempo dal verificarsi di un evento**
- **tempo limitato superiormente**
- **controllo di sistemi critici (es. impianti chimici, navigazione aerea o spaziale)**

## Sistemi fault-tolerant

- **in grado di continuare a funzionare anche in presenza di guasti**
- **degrado graduale delle prestazioni**
- ***ridondanza* hw e/o sw**
- ***hot-fix***

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Interfaccia utente

**Interfaccia testuale:**

- **interprete dei comandi ( *shell* )**

**Interfaccia grafica (a finestre):**

- **finestra = output di un processo**
- **finestra attiva ( *focus* ) = input**
- **paradigma WIMP = Window**
  **Icon**
  **Mouse**
  **Pointer**

## Cosa succede all'accensione di un elaboratore?

**Il processo di *bootstrap*:**

- **programma in ROM per l'auto-test dell'hardware**
- **programma in ROM per caricare da disco (o da rete) il *boot program***
- **il programma di boot carica il sistema operativo**
- **il S.O. assume il controllo dell'elaboratore**

## MS-DOS

- **CPU Intel 8086 (16 bit) e successive**
- **monotask**
- **monoutente**
- **file-system gerarchico (nomi 8.3)**
- **memoria limitata (1 MB / 640 KB)**
- **nessuna protezione**
- **PC- / IBM- / DR-DOS**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## OS/2

- **multitask**
- **monoutente**
- **HPFS (High Performance FS)**
- **PM (Presentation Manager)**
- **pensato per 80286 (16 MB RAM), ma arrivato tardi!**
- **la versione per 80386 non ha avuto maggior successo**

## Windows-NT / 2000 / XP / 2003 / Vista

- **CPU Intel 80386/486/Pentium**
- **multitask**
- **multiutente**
- **NTFS (NT File System)**
- **microkernel, thread**
- **sistema a 32 bit (64 bit da XP)**

## UNIX

- **nato negli anni ‘60 (AT&T Bell Labs)**
- **rimasto all’avanguardia perchè sviluppato nelle università (UCB)**
- **multitask**
- **multiutente**
- **ottima integrazione in rete**
- **portabilità dei programmi**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## UNIX: dialetti e standard

**Ogni ditta ha il "suo" UNIX: Solaris, AIX, HP-UX, TruUnix64, ...**

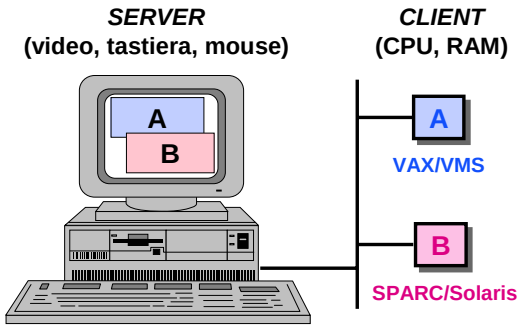
**Standard:**

- **POSIX (IEEE, DOD)**
- **OpenGroup = X/OPEN + OSF (SPEC-1170, XPG, COSE, ...)**
- **open source / free: Linux, FreeBSD, NetBSD, OpenBSD, …**

## Il sistema X-windows

- **sistema grafico, a finestre, distribuito in rete eterogenea**
- **è il prodotto più famoso del progetto Athena del MIT**
- **anche noto come X11**
- **è il sistema a finestre nativo di Unix & C.**

## Architettura di X-Windows




*© A.Lioy - Politecnico di Torino (1992-2008)*

## VMS, OpenVMS

- **sistema proprietario DEC/COMPAQ/HP**
- **CPU DEC Vax o AXP**
- **molto robusto**
- **penalizza un po' le prestazioni**
- **cluster**
- **ne esiste una versione real-time**

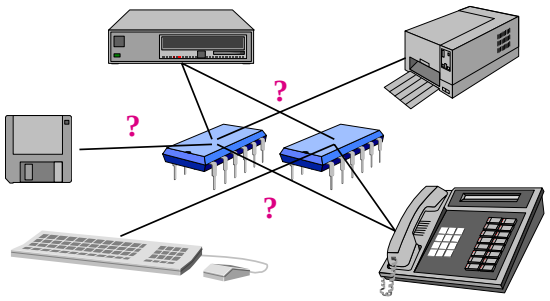
## Sistemi operativi IBM

**Ambienti operativi per mainframe:**

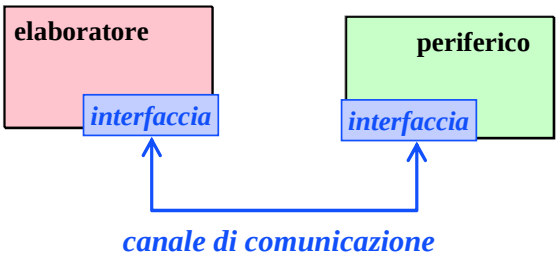
- **TSO (Time-Sharing Option) su MVS**
- **CMS (Conversational Monitor System) su VM (Virtual Machine)**
- **CICS monitor transazionale**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Collegamento di dispositivi periferici




## Collegamento di principio




## Gestione delle interfacce

***Interrupt*** **(priorità):**

- **un'interfaccia richiede il servizio quando ha dei dati da elaborare**
- **complesso, ma la CPU può fare altro**

***Polling*** **(busy-wait, round-robin):**

- **la CPU domanda periodicamente ad ogni interfaccia se ha dei dati da elaborare**
- **semplice, ma occupa la CPU**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Trasferimento dati tra periferici e CPU (memoria)

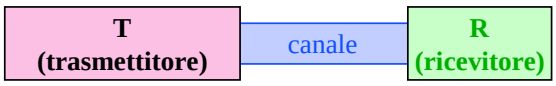
**I dati da / per i periferici sono memorizzati in RAM e devono essere trasferiti (celermente).**

**Soluzione 1 = la CPU trasferisce i dati**

**Soluzione 2 = DMA (Direct Memory Access)**

- **la CPU indica la quantità di dati, la zona di memoria ed il periferico**
- **il trasferimento avviene direttamente tra periferico e RAM, senza intervento della CPU (che può svolgere altre operazioni)**

## Comunicazione tra apparecchiature elettroniche

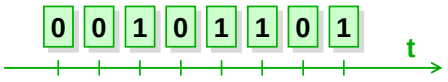



***velocità di trasmissione*** ***capacità del canale*** ***velocità di ricezione***

**bit/s (BPS, Bit Per Second)**

## Comunicazione seriale

**Dati trasmessi uno per volta in sequenza ordinata:**

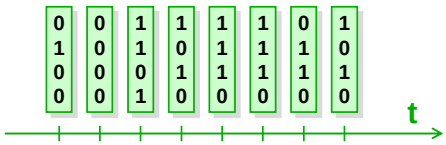



+ **collegamento semplice**
– **comunicazione lenta**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Comunicazione parallela

**Più dati trasmessi simultaneamente:**




+ **comunicazione veloce**
– **collegamento complesso**
– **possibile interferenza tra i dati**

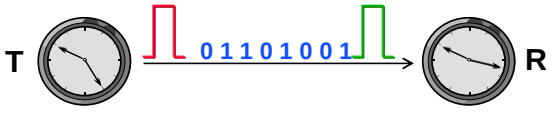
## Comunicazione sincrona

**I due apparati possiedono uno stesso riferimento temporale con cui delimitano la durata dei dati:**

+ **comunicazione veloce**
– **sincronizzazione dei riferimenti**

## Comunicazione asincrona

**I due apparati sincronizzano lo scambio dei dati mediante speciali segnali detti *segnali di controllo* (1/2 start bit, 0/1 stop bit):**




+ **nessun riferimento temporale comune**
– **comunicazione lenta**

© A.Lioy - Politecnico di Torino (1992-2008)

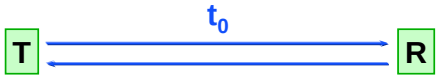
## Comunicazione unidirezionale (half-duplex)

**Ad ogni istante di tempo i dati viaggiano in una sola direzione:**

+ **collegamento semplice**
– **comunicazione bidirezionale impossibile o lenta (inversione del flusso)**

## Comunicazione bidirezionale (full-duplex)

**Ad ogni istante di tempo è sempre possibile uno scambio simultaneo di dati:**




+ **comunicazione veloce**
– **collegamento complesso**

## Linea seriale

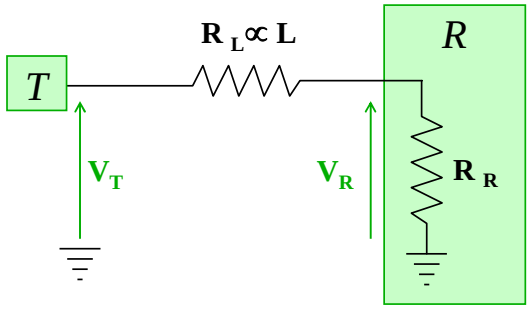
- **cavi in rame schermati**
- **trasmette una tensione elettrica**
- **necessita di una massa comune (perché la tensione elettrica è in realtà una differenza di potenziale elettrico)**
- **trasmissione asincrona**
- **due standard (RS-232, RS-423)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Linea seriale

- **numero minimo di segnali**
  **half-duplex = 2 ( TxD, Gnd )**
  **full-duplex = 3 ( TxD, RxD, Gnd )**
- **velocità multipla di 300 bit/s**
- **velocità tipica 1200 / 9600 / 19200**

## Schema di una linea elettrica (a bassa frequenza)




## Limiti della linea seriale

$$V_R = V_T \cdot \frac{R_R}{R_L + R_R}$$

- **massima distanza < 100 m**
- **velocità ∝ 1/L (300 ... 38,400 bit/s)**
- **velocità tipica 9600 bit/s**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Standard per linea seriale

**Lo standard più diffuso è RS-232 (standard CCITT V.24):**

- **9 poli**
- **25 poli**

**In alternativa RS-423:**

- **6 poli**

## RS-232

- **zero logico > + 3 V**
- **uno logico < - 3 V**
- **segnali più importanti:**

| | |
|---|---|
| TxD | ( Transmitted Data ) |
| RxD | ( Received Data ) |
| Gnd | ( Ground ) |
| DTR | ( Data Terminal Ready ) |
| DSR | ( Data Set Ready ) |
| RTS | ( Request To Send ) |
| CTS | ( Clear To Send ) |

## Collegamento RS-232

*© A.Lioy - Politecnico di Torino (1992-2008)*

## RS-423

- **parzialmente compatibile con RS-232**
- **segnali:**
  - **TD**
  - **RD**
  - **TD-GND**
  - **RD-GND**
  - **DTR**
  - **DSR**

## Linea parallela

- **cavi in rame schermati**
- **trasmette una tensione elettrica**
- **necessita di una massa comune**
- **da 8 a 64 bit per dato**
- **tre standard per scopi diversi (SCSI, Centronics, IEEE-488/HP-IB)**
- **usabile solo su distanze brevi ( ~ 10 m)**

## Lo standard Centronics

- **linea parallela da 8 bit (+controlli)**
- **sincrona**
- **unidirezionale**
- **collegamenti punto-punto**
- **per collegare stampanti (veloci)**
- **oggi evoluto a IEEE-1284**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## IEEE 1284

- **estensione dello standard Centronics**
- **bidirezionale, max 30 ft**
- **channel addressing (es. fax/printer/modem)**
- **ECP (Enhanced Capability Port) mode:**
  - **compressione dei dati + code (FIFO)**
  - **per stampanti e scanner**
  - **2 … 4 MB/s**
- **EPP (Enhanced Parallel Port) mode:**
  - **per CD-ROM, hard-disk, …**
  - **0.5 … 2 MB/s**

## Lo standard IEEE-488 (HP-IB)

- **linea parallela da 8 bit**
- **16 fili (8 dati, 3 handshake, 5 bus management) + terra**
- **sincrona**
- **bidirezionale**
- **organizzazione a bus**
- **velocità max 8 Mb/s**
- **soprattutto per strumentazione automatica (un tempo anche per periferici)**

## Lo standard SCSI (Small Computer System Interface)

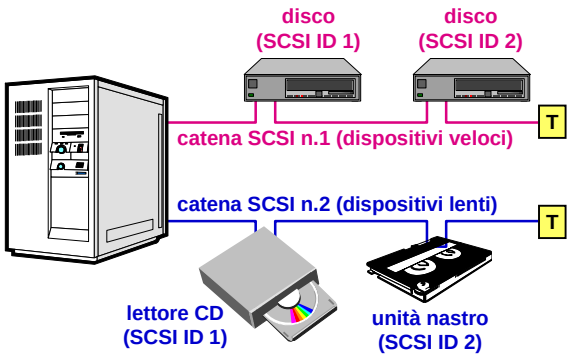
- **standard ANSI**
- **bus bidirezionale:**
  - **8 bit (narrow SCSI)**
  - **16 bit (wide SCSI)**
- **collegamento logico: bus**
- **collegamento fisico: cascata (max 8 o 16 dispositivi per catena)**
- **per dischi, dispositivi di memoria o periferici veloci**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## SCSI – trasferimento dati

- **trasferimento asincrono o sincrono (3x); es:**
  - **SCSI-1 = asinc 1.5 MB/s, sinc 5.0 MB/s**
  - **SCSI-2 = asinc 3.0 MB/s, sinc 10.0 MB/s**
- **attenzione: sincronismo sul periferico più lento!**
- **SCSI normale (single-ended): 1 filo/segnale**
- **SCSI differenziale: 2 fili/segnale e si prende la differenza (= meno sensibile ai disturbi, cavi più lunghi – max 25 m carico singolo, 12 m carico multiplo)**

## Catene ed identificatori SCSI




## Ultra SCSI

- **versione più moderna**
- **solo differenziale**
- **Ultra-3 solo wide**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Evoluzione SCSI

| | *distanza [ m ]* | *velocità [ MB/s ]* | *dispos. [ n ]* |
|---|---|---|---|
| SCSI-1 | 6 | 5 | 8 |
| SCSI-2 | 6 | 5-10 | 8 / 16 |
| SCSI-2 fast | 3 | 10-20 | 8 |
| SCSI-2 wide | 3 | 20 | 16 |
| SCSI-2 fast wide | 3 | 20 | 16 |
| Ultra SCSI-3 | 1.5 | 20 | 8 |
| Ultra SCSI-3 wide | 1.5 | 40 | 16 |
| Ultra-2 SCSI | 12 | 40 | 8 |
| Ultra-2 SCSI wide | 12 | 80 | 16 |
| Ultra-3 (Ultra160) SCSI | 12 | 160 | 16 |
| Ultra-3 (Ultra320) SCSI | 12 | 320 | 16 |

## IDE (Integrated Drive Electronics)

- **interfaccia per dischi**
- **max 2 dischi per canale (master e slave)**
- **max dimensione dischi = 504 MB (limite del BIOS)**
- **max velocità = 4 MB/s**
- **standard ANSI:**
  **ATA (Advanced Technology Attachment)**

## Fast / Ultra ATA (EIDE)

- **Fast ATA = 11 MB/s**
- **ATA-2 = 16 MB/s, dischi max 8.4 GB**
- **Ultra ATA (o ATA-4) = 33 MB/s**
- **Ultra ATA/66 (o ATA-5) = 66 MB/s**
- **ATA-6 = 100 MB/s**
- **EIDE (Enhanced IDE) = ATA ≥ 2**
- **a partire da ATA-4 sono disponibili i comandi ATAPI (ATA Packet Interface) per collegare anche dispositivi diversi dai dischi (es. nastri, CD-ROM, …)**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## SATA (serial ATA)

- **trasmissione seriale (invece che parallela) con linea dedicata**
- **cavo più sottile, flessibile e lungo**
- **7 fili (4 segnali + 3 ground) contro 40/7G fili in PATA (80/40G da ATA-3); ciò diminuisce:**
  - **influenza reciproca tra segnali adiacenti (crosstalk)**
  - **interferenze elettromagnetiche con altri dispositivi (EMI)**
- **throughput base = 150 MB/s**
- **pianificata evoluzione sino a 600 MB/s**

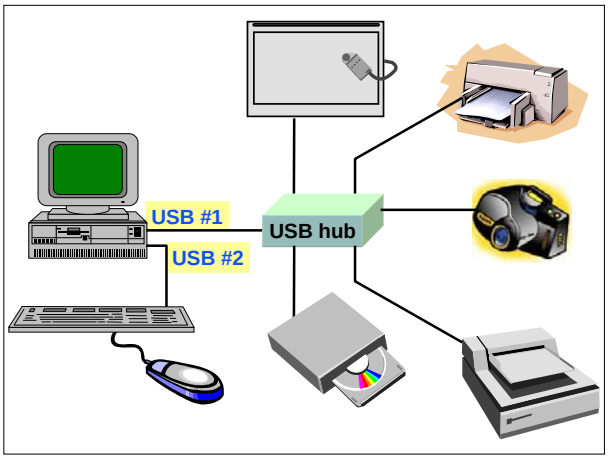
## USB (Universal Serial Bus)

- **trasmissione seriale, half-duplex**
- **polling comandato dal computer**
- **collegamento logico a bus, max 127 dispositivi, alimentabili dal bus**
- **sistema plug-and-play**
- **USB 1.0 = low speed (1.5 Mbps)**
- **USB 1.1 = medium speed (1.5…12 Mbps)**
- **USB 2.0 high speed (160…480 Mbps = 20…60 MB/s)**

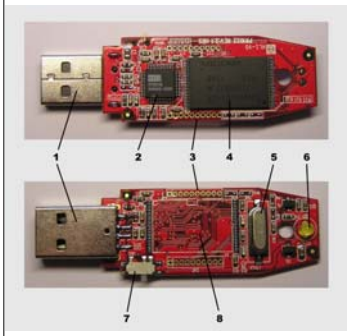
## USB: cablaggio

- **cavo a 4 fili con connettore piatto (2 per dati, 2 per alimentazione a 5 V)**
- **lunghezza max cavo = 5 m**
- **hub per collegare più dispositivi**
- **collegamento in cascata di massimo 5 hub (quindi distanza max = 30 m)**

*© A.Lioy - Politecnico di Torino (1992-2008)*




## USB memory pen




1. connettore USB
2. controllore USB
3. punti di test
4. memoria flash
5. oscillatore al quarzo
6. LED di attività
7. read-only switch
8. spazio per una seconda memoria flash

*( nota: immagine tratta da Wikipedia )*

## IEEE-1394

- **Firewire (Apple)**
- **i.LINK (Sony)**
- **standard de facto per collegamenti digitali (non analogici!) tra dispositivi video: videocamere, videoregistratori, DVD, sistemi di editing video, PC, …**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## IEEE 1394 - caratteristiche

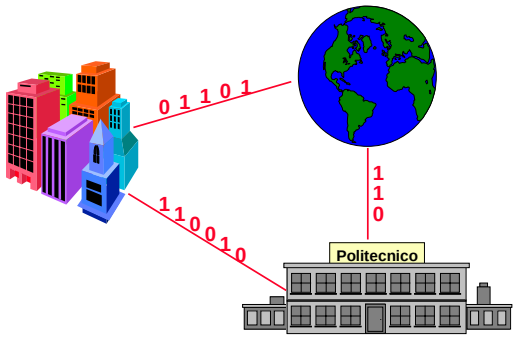
- **trasferimento asincrono o isocrono**
- **velocità:**
  - **100 / 200 / 400 Mbps (sul cavo, peer-to-peer)**
  - **12.5 / 25 / 50 Mbps (dentro al computer)**
- **peer-to-peer (direttamente tra dispositivi)**
- **collegamento logico a stella, massimo 63 dispositivi**
- **lunghezza massima cavo = 4.5 m**
- **collegamento in cascata di massimo 16 dispositivi (quindi distanza max = 72 m)**

## Equilibrio e bilanciamento

- **le prestazioni globali di un sistema di elaborazione richiedono che tutte le sue componenti siano equilibrate**
- **il cammino dei dati da / verso la CPU non deve incontrare colli di bottiglia**
- **esempio: inutile comprare disco SCSI Ultra160 con CPU lenta o bus PCI piccolo, lento o affollato**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Trasmissione dati




## Problematiche di trasmissione dati

- **per distanze superiori a qualche decina di metri, occorrono apparati di telecomunicazione**
  **= collegamento fisico**
- **poiché i sistemi coinvolti possono essere di tipo diverso, occorre definire degli standard per lo scambio e l'interpretazione dei dati**
  **= collegamento logico (protocolli, reti)**

## Collegamenti fisici

- **linea punto-punto dedicata (CDN)**
  - **O(10 Mbps)**
  - **canone annuale O(50 K€)**
- **linea punto-punto commutata (= linea telefonica)**
  - **da 9,600 ... 56 Kbps a O(1 Mbps)**
  - **tariffazione a tempo**
  - **commutazione di circuito**
  - **modem analogico, ISDN**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Reti locali / geografiche

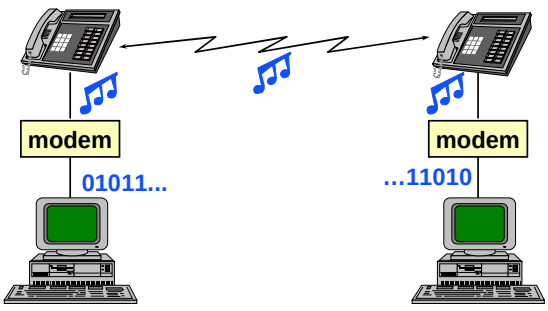
**LAN**
- **Local Area Network**

**MAN**
- **Metropolitan Area Network**

**WAN**
- **Wide Area Network**

**Tecnologie: Ethernet, Frame Relay, ATM, Sonet/SDH, …**

## Modem (modulatore - demodulatore)




## Modem

- **apparato di telecomunicazione atto a scambiare dati digitali**
- ***modem in banda fonica* (o traslata)**
  **opera su linea telefonica commutata (300 ... 3400 Hz)**
- ***modem in banda base***
  **opera solo su circuiti diretti**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Standard ITU per modem in banda traslata

| *standard* | *bps* | *linea* |
|---|---|---|
| V.21 | 300 | commutata |
| V.22 | 1,200 | comm. / dedic. |
| V.22bis | 2,400 | comm. / dedic. |
| V.26 | 2,400 | dedicata |
| V.26bis | 2,400 | commutata |
| V.27 | 4,800 | dedicata |
| V.27ter | 4,800 | commutata |
| V.29 | 9,600 | dedicata |
| V.32 | 9,600 | comm. / dedic. |
| V.33 | 14,400 | dedicata |

## Standard ITU per modem in banda traslata

| *standard* | *bps* | *linea* |
|---|---|---|
| V.34 | 38,400 | commutata |
| V.90 | 57,600 | comm. / dedic. |

velocità diversa in trasmissione e ricezione se la linea è analogica

## ISDN

- **Integrated Services Digital Network**
- **canale digitale (= linea ISDN)**
  - **non necessario modem**
  - **necessario telefono / scheda ISDN**
- **2 o 3 canali digitali:**
  - **canale dati n.1 (64 Kbps)**
  - **canale dati n.2 (64 Kbps)**
  - **canale ausiliario (16 Kbps), solitamente dedicato alla fonia o alle segnalazioni ausiliarie**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## ADSL

- **Asymmetric Digital Subscriber Line**
- **trasmissione digitale sul normale doppino telefonico (tramite modem/router speciale)**
- **3 canali:**
  - **download dati (tra 2 e 6 Mbps)**
  - **bidirezionale dati (max 640 Kbps)**
  - **bidirezionale voce (per backup, nel caso si guasti quello dati su cui normalmente viene fatta passare anche la voce); potrebbe essere ISDN**

## Ethernet e WiFi

**Ethernet:**

- **standard per reti locali cablate (IEEE 802.3)**
- **10 Mbps (originale), 100 Mbps, 1000 Mbps (anche detta Gigabit Ethernet)**

**WiFi (= Wireless Fidelity):**

- **standard per reti locali radio (IEEE 802.11)**

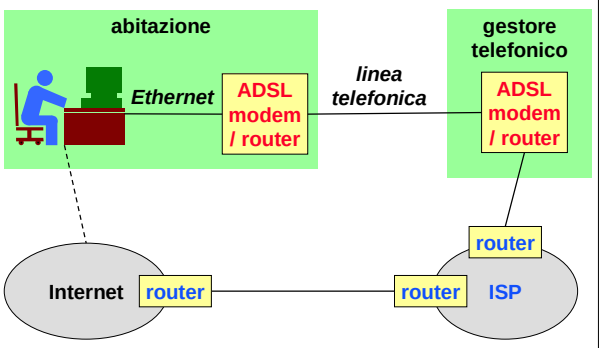
| | *velocità* | *freq.* | *compat.* |
|---|---|---|---|
| 802.11b | 11 Mbps | 2.4 GHz | b |
| 802.11a | 54 Mbps | 5.0 GHz | a |
| 802.11g | 54 Mbps | 2.4 GHz | b, g |
| 802.11n | 100 Mbps | 2.4 GHz | b, g, n |

## La rete Internet

- **è una rete di reti**
- **i nodi della rete (router) si scambiano i dati fino a farli arrivare a destinazione**
- **un canale (o link) tra due router può trasmettere simultaneamente i dati di più utenti sino a saturare la capacità del canale**
- **se più utenti usano simultaneamente un canale, la sua capacità viene divisa equalmente tra tutti gli utenti**
- **NOTA: modello molto semplificato**

*© A.Lioy - Politecnico di Torino (1992-2008)*

## Accesso ad Internet via ADSL




## Esercizio

**Due utenti sono collegati via ADSL 2 Mbps alla stessa centrale telefonica, a sua volta collegata ad Internet tramite una linea a 10 Mbps.**

**Calcolare il tempo complessivo affinché ciascun utente scarichi un file da 100 MB.**

**Soluzione:**

- **ciascun utente accede ai dati a 2 Mbps**
- **8 x 100 Mbit / 2 Mbit/s = 400 s = 6' 40"**
- **tempo totale = 6' 40" (perché i due utenti operano in parallelo)**

## Esercizio

**Due utenti sono collegati via ADSL 2 Mbps alla stessa centrale telefonica, a sua volta collegata ad Internet tramite una linea a 2 Mbps.**

**Calcolare il tempo complessivo affinché ciascun utente scarichi un file da 100 MB.**

**Soluzione:**

- **ciascun utente accede ai dati a 1 Mbps (perché il collegamento centrale-Internet è usato al 50% da ciascun utente)**
- **8 x 100 Mbit / 1 Mbit/s = 800 s = 13' 20"**
- **tempo totale = 13' 20"**

*© A.Lioy - Politecnico di Torino (1992-2008)*

**Tabella dei primi 128 codici ASCII (codifica decimale ed esadecimale)**

| *dec* | *hex* | *char* | *dec* | *hex* | *char* | *dec* | *hex* | *char* | *dec* | *hex* | *char* |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | NUL | 32 | 20 | *spazio* | 64 | 40 | @ | 96 | 60 | ‘ |
| 1 | 1 | SOH | 33 | 21 | ! | 65 | 41 | A | 97 | 61 | a |
| 2 | 2 | STX | 34 | 22 | " | 66 | 42 | B | 98 | 62 | b |
| 3 | 3 | ETX | 35 | 23 | # | 67 | 43 | C | 99 | 63 | c |
| 4 | 4 | EOT | 36 | 24 | $ | 68 | 44 | D | 100 | 64 | d |
| 5 | 5 | ENQ | 37 | 25 | % | 69 | 45 | E | 101 | 65 | e |
| 6 | 6 | ACK | 38 | 26 | & | 70 | 46 | F | 102 | 66 | f |
| 7 | 7 | BEL | 39 | 27 | ’ | 71 | 47 | G | 103 | 67 | g |
| 8 | 8 | BS | 40 | 28 | ( | 72 | 48 | H | 104 | 68 | h |
| 9 | 9 | HT | 41 | 29 | ) | 73 | 49 | I | 105 | 69 | i |
| 10 | A | LF | 42 | 2A | * | 74 | 4A | J | 106 | 6A | j |
| 11 | B | VT | 43 | 2B | + | 75 | 4B | K | 107 | 6B | k |
| 12 | C | NP | 44 | 2C | , | 76 | 4C | L | 108 | 6C | l |
| 13 | D | CR | 45 | 2D | - | 77 | 4D | M | 109 | 6D | m |
| 14 | E | SO | 46 | 2E | . | 78 | 4E | N | 110 | 6E | n |
| 15 | F | SI | 47 | 2F | / | 79 | 4F | O | 111 | 6F | o |
| 16 | 10 | DLE | 48 | 30 | 0 | 80 | 50 | P | 112 | 70 | p |
| 17 | 11 | DC1 | 49 | 31 | 1 | 81 | 51 | Q | 113 | 71 | q |
| 18 | 12 | DC2 | 50 | 32 | 2 | 82 | 52 | R | 114 | 72 | r |
| 19 | 13 | DC3 | 51 | 33 | 3 | 83 | 53 | S | 115 | 73 | s |
| 20 | 14 | DC4 | 52 | 34 | 4 | 84 | 54 | T | 116 | 74 | t |
| 21 | 15 | NAK | 53 | 35 | 5 | 85 | 55 | U | 117 | 75 | u |
| 22 | 16 | SYN | 54 | 36 | 6 | 86 | 56 | V | 118 | 76 | v |
| 23 | 17 | ETB | 55 | 37 | 7 | 87 | 57 | W | 119 | 77 | w |
| 24 | 18 | CAN | 56 | 38 | 8 | 88 | 58 | X | 120 | 78 | x |
| 25 | 19 | EM | 57 | 39 | 9 | 89 | 59 | Y | 121 | 79 | y |
| 26 | 1A | SUB | 58 | 3A | : | 90 | 5A | Z | 122 | 7A | z |
| 27 | 1B | ESC | 59 | 3B | ; | 91 | 5B | [ | 123 | 7B | { |
| 28 | 1C | FS | 60 | 3C | < | 92 | 5C | \ | 124 | 7C | \| |
| 29 | 1D | GS | 61 | 3D | = | 93 | 5D | ] | 125 | 7D | } |
| 30 | 1E | RS | 62 | 3E | > | 94 | 5E | ^ | 126 | 7E | ~ |
| 31 | 1F | US | 63 | 3F | ? | 95 | 5F | _ | 127 | 7F | DEL |

**Nomi dei caratteri di controllo**

| | | | |
|---|---|---|---|
| ACK | Acknowledgment | EOT | End of Transmission |
| BEL | Bell | ESC | Escape |
| BS | Backspace | ETB | End of Transmission Block |
| CAN | Cancel | ETX | End of Text |
| CR | Carriage Return | FF | Form Feed (o New Page, NP) |
| DC1 | Device Control 1 (o XON) | FS | File Separator |
| DC2 | Device Control 2 | GS | Group Separator |
| DC3 | Device Control 3 (o XOFF) | HT | Horizontal Tab |
| DC4 | Device Control 4 | LF | Line Feed (o New Line, NL) |
| DEL | Delete | NAK | Negative Acknowledgement |
| DLE | Data Link Escape | NUL | Null |
| EM | End of Medium | SI | Shift In |
| ENQ | Enquiry | SO | Shift Out |
| SOH | Start of Header | STX | Start of Text |
| SUB | Substitute | SYN | Synchronous Idle |
| US | Unit Separator | VT | Vertical Tab |

*© A.Lioy - Politecnico di Torino (1992-2008)*